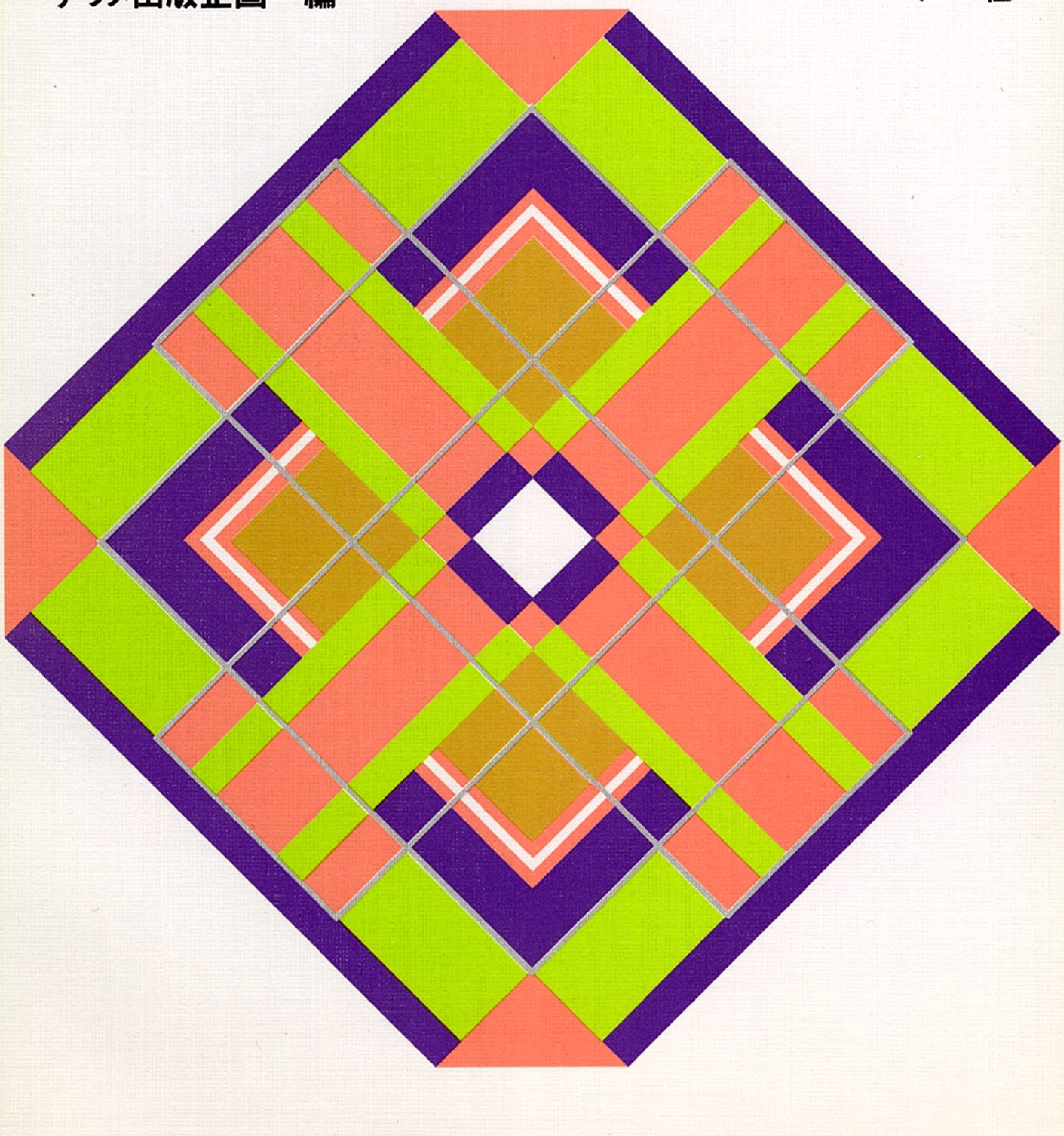
Personal Computer

PC-8801 NEC mkII FR|MR

データファイル入門[2]

ナツメ出版企画＝編

ナツメ社



Personal Computer
PC-8801mkII FR|MR
データファイル入門[2]
ナツメ出版企画＝編
ナツメ社

[本書を読まれる方へ]

⊙本書は、前に発行した
『PC-8801MKⅡFR|MRデータファイル入門[1]』
の続編にあたります。
⊙BASICをひととおり学び、
いざ実用化という段になって
必ず突き当たるのがデータの扱いです。
そこで、どうしてもマスターしなければ
ならないのがディスク操作です。
⊙ディスクに関する本は
たくさん発行されていますが、
具体例にそくしたものが少なく、
自分の仕事にどう活用するかとなると
どうしても行き詰まってしまうようです。
⊙そこで本書では、
前編で扱ったテーマとも関連性をもたせ、
みなさんが知らず知らずのうちに、
より高いレベルへ進めるよう工夫しました。
またPC-8801FR|MRでは
N88日本語BASICがソフトでサポートされ、
とくに漢字の扱いが大変やりやすくなりました。
そこで前編では触れることのできなかった
漢字データ処理について、
巻末で、住所録を例にとって解説しました。
⊙PC-8801MKⅡFR|MRは、
みなさんの努力次第で
ずいぶん有効に使えるはずです。
本書でのヒントをもとに
ぜひビジネスに活用されることを期待します。

cover design——nobuo nakagaki+yoshifumi hayase

Personal Computer

PC-8801

NEC

mkII FR/MR

データファイル入門[2]

ナツメ出版企画＝編

ナツメ社

この本を読まれる方へ

この本は、前に発行した

『PC-8801MKⅡFR／MRデータファイル入門１』

の続編にあたるものです。

したがって、ディスク操作の基本的なコマンド、ステートメントなどについては、いちいち解説を加えておりませんので、これからディスクの勉強を始めようという方は、まず、この本の前編をお読みください。そして、概論をマスターしたのち、この本に進めば、なんら抵抗なく進めると思います。

本書、および前編とも、テーマには日常の身近な例を取り上げていますので、ある程度親しみを持って進んでいただけるのではないかと考えております。ただ、ここで取り上げたプログラムが、そのままあなたの実務に適用できるということでは必ずしもなく、本書の目的が、むずかしいといわれているディスク操作の応用のしかたを具体例にそくしてやさしく解説する点にありますので、本書でのプログラムを変形したり、あるいはこれを参考に自分専用のシステムを作ったりするのは、あなた次第ということになります。

パソコンをワープロに使いたいとか、あるいは販売管理に使いたいとか、みなさんの目的はまちまちだと思います。そして市販のソフトにもかなり便利なものがありますから、ある部分はこういったソフトに頼ることもできます。

しかし、これらのソフトは特殊化されたものではなく一般的な場合を想定していますから、こんなこともやってみたい、あんなこともやってみたいという希望があっても、すべてをかなえてくれるというわけにはまいりません。

これがよく「市販のビジネスソフトを買ったが、かゆいところに手がとどかない」といわれるゆえんです。

ですから理想的なのは、市販のソフトに頼る部分、自分でプログラムを作って活用する部分というように、自分の仕事をある程度仕分けして柔軟な態度で望むことです。

市販のソフトでも適用できそうなものがあれば、どんどん利用したほうが労力の点でもずいぶん節約できるはずです。しかしこの場合大切なことは、すでに自分でディスク操作ができプログラムが組めるということは、あなたには大変なプラスとなって返ってくるでしょう。

話が横道にそれましたが、PC-8801MKIIFR／MRは、あなたの使い方次第によっては大変な力を発揮します。記憶容量も大きいですし、処理スピードも非常に向上しています。あなたのやってみたい仕事をぜひプログラム化し、PC-8801MKIIFR／MRに処理させてみてください。この本がお手伝いします。

ところで、PC-8801MKIIFR／MRではN88日本語BASICによる漢字の扱いができますが、データファイルの操作をもう少し覚えていただきたいので、第１章～第３章ではN88BASICモードのもとで、カタカナ・英数字によりプログラムを組んでいきます。

そして第４章で、N88日本語BASICを起動して漢字版の住所録プログラムを作ってみます。

どうか途中でめげずに、最後までがんばってください。

なお本書を終えてから、もう少し実用的なプログラム例を学習してみたい、あるいは使ってみたいという向きには次の書籍をおすすめします。

PC-8801MKIIFR／MR　ディスク実用プログラム集

パーソナルデータベース、顧客管理、成績管理など、大変便利なプログラム15本とその使い方が掲載されています。

また上記書籍の全プログラムを収めた別売ディスケットサービスも行われています。時間のない方、早く使いたいという方はぜひ利用されるとよいでしょう。

もくじ

第1章

セクタの もう1つの使い方

1·1 ファイル内でのデータ設計

プログラミングに入る前に

私たちは、この本の前編

『PC-8801MKⅡFR／MRデータファイル入門1』

で、シーケンシャルファイルおよびデータファイルについてのひととおりの知識を学んできました。

あとでふりかえってみますと、こんなことも付け加えておけば、もっと理解の助けになったのではないか、と思われる点も多少ありますが、一応編集部からの要望「ナツメ社パソコンシリーズの方針」つまり**「パソコンの実用化」**にそくして解説しましたので、具体的なアウトラインはご理解いただけたのではないかと考えております。

BASICをひととおり学び、いざ実用化という段階にきて必ず突き当たるのがデータの扱いです。そこで、どうしてもマスターしなければならないのが、**DISK-BASIC**です。

そしてPC-8801MKⅡFRモデル30、MRには2ドライブのディスク装置が内蔵されています。使う使わないの問題ではなく、ディスク操作をマスターせざるを得ないといったほうがよいでしょう。

仕事に熱意をもっている人たちが、せっかく購入したパソコンをなんとか実用化したいと考えるのは人情です。パソコン活用が軌道にのったからといって決して楽ができるわけではありません。確かに、現在の事務処理その他については、ずいぶん能率化がはかれます。しかし、その先にまた次なる問題がひかえています。そして、この新しい問題とかテーマにチャレンジしていこうと意欲を燃やすところに、いわゆる「ヤル気のある人間」の「真実」があるといえますし、一社会人としての進歩が期待できるのです。

ある問題が解決してしまったら、もうそれでおしまいというのでは、これは「ごくあたり前の人間」です。

この本を読んで、何が何でもパソコンを実用化したいと考えているあなたがたは、おそらく大きな欲望をもっていらっしゃると思います。そして、この事実を前提に、筆者も筆を進めていきたいと思います。

プログラムも白紙の状態から作っていきますので、ずいぶん曲りくねった道をたどることも予想されますし、完成したからといって、いくつかバグもでてくるかもしれません。万一そんなことがありましたら一緒に解決していきましょう。

プログラムで、いつも決まりきった仕事をする部分をサブルーチンといいましたね。プログラムでしたらルーチン化もわかりますが、私たちは、ぜひ「ルーチン化した人間」にならないようがんばろうではありませんか。

ずいぶん話が脱線してしまいましたが、結論は、実用化するならディスクを使わないと話にならないということです。

最初の本を読んでみなさんはどう感じられたかしれませんが、初めのうちは、ディスクというのはとにかくとっつきずらいものです。とくにデータがどんなふうに書き込まれているのか見えませんしね。しかし、試行錯誤を重ねているうちに徐々になれてくると思います。

ディスクを扱うようになりますと、最初は**シーケンシャルファイル**から入ります。そこで、**ランダムファイル**よりシーケンシャルファイルのほうが理解しやすいというので、なんでもシーケンシャルファイルでかたづけようとする人がいます。しかし、これでは、仕事を型にはめてしまうようなもので、発展性がありません。なんとしても、ランダムファイルを自由に扱えるようになりたいものです。なぜなら、**ランダムファイルには非常にフレキシブルな面があります**し、慣れてしまえばそれほどむずかしくはないからです。

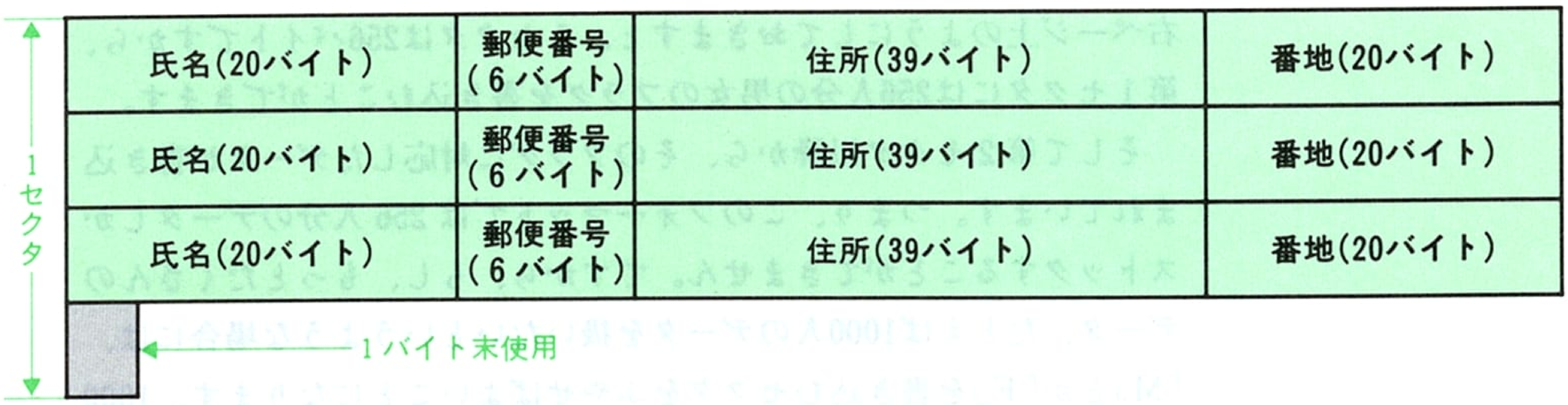
ちょっと違ったデータ設計

前編において私たちは、データファイルの1セクタ内に1レコードを書き込むことをまず学び、続いて1セクタ内に3レコードを割り当て、ディスクをさらに有効に利用することを知りました。

そのときのデータ設計は、およそ下の図のようでした。**氏名、郵便番号、住所、番地を1レコードとし、計3人分のデータを1セクタ内に書き込んだ**わけです。未使用は1バイトですから、大変効率的にディスクを使うことができました。

レコード
●小さな項目の集まりのことをレコードといいます。

この章では、考え方をもう少し広げて、ちょっと違ったデータ設計をしてみることにしましょう。



この章で扱うテーマですが、話に連続性をもたせたいことと、そのほうが理解しやすいというメリットから、前回に引き続き同じ宛て名書きを選ぶことにします。

あなたがたが実用化したいというテーマはまちまちでも、データ設計の基本は共通していますので、この点をご理解ください。

私たちは前回は、**3レコードを1セットとして第1セクタに書き込んでいました。**しかし、ここでは、**第1セクタはほかのデータを書き込むためにあけておき、氏名、郵便番号などのデータは第2セクタから書き込む**ことにします。

では、第1セクタを何のために利用するかといいますと、男性向けの宛て名、女性向けの宛て名を区別したいので、その性別を書き込むのです。

たとえば、1番目のレコードの人が男性であったら、第1セクタの1バイト目に「M」を、2番目の人が女性であったら、第1セクタの2バイト目に「F」を書き込む、というようにするのです。**〈注〉**

こうしておくと、もし男性だけの宛て名書きが必要な場合、ファイルをOPENして、第1セクタのデータをチェックし、「M」という書き込みのあるものだけを抜きだし、次に、ここでわかった順位をもとに、第2セクタ以降から該当する位置をGETすれば、男性だけの宛て名を得ることができます。**〈注〉**

つまり、前回のファイル内でのデータ設計は、第1セクタから最終セクタまで、すべて同じ**フォーマット**になっていましたが、ここでは第1セクタだけを異なるフォーマットで使用しようというわけです。

この様子を図に示すと、次ページの図のように表現できます。

ここで登場願った方々も、前回と同じようにだいぶ古い人たちですが、この原稿を書いている人が古いのでご容赦ください。とにかく理屈を理解していただくことが目的ですから、よろしくお願いします。

フラグ

〈注〉Mとか Fのことをフラグと呼ぶことにします。フラグは英語で「旗」のことです。つまり「M」という旗なら男性、「F」という旗なら女性と考えるのです。

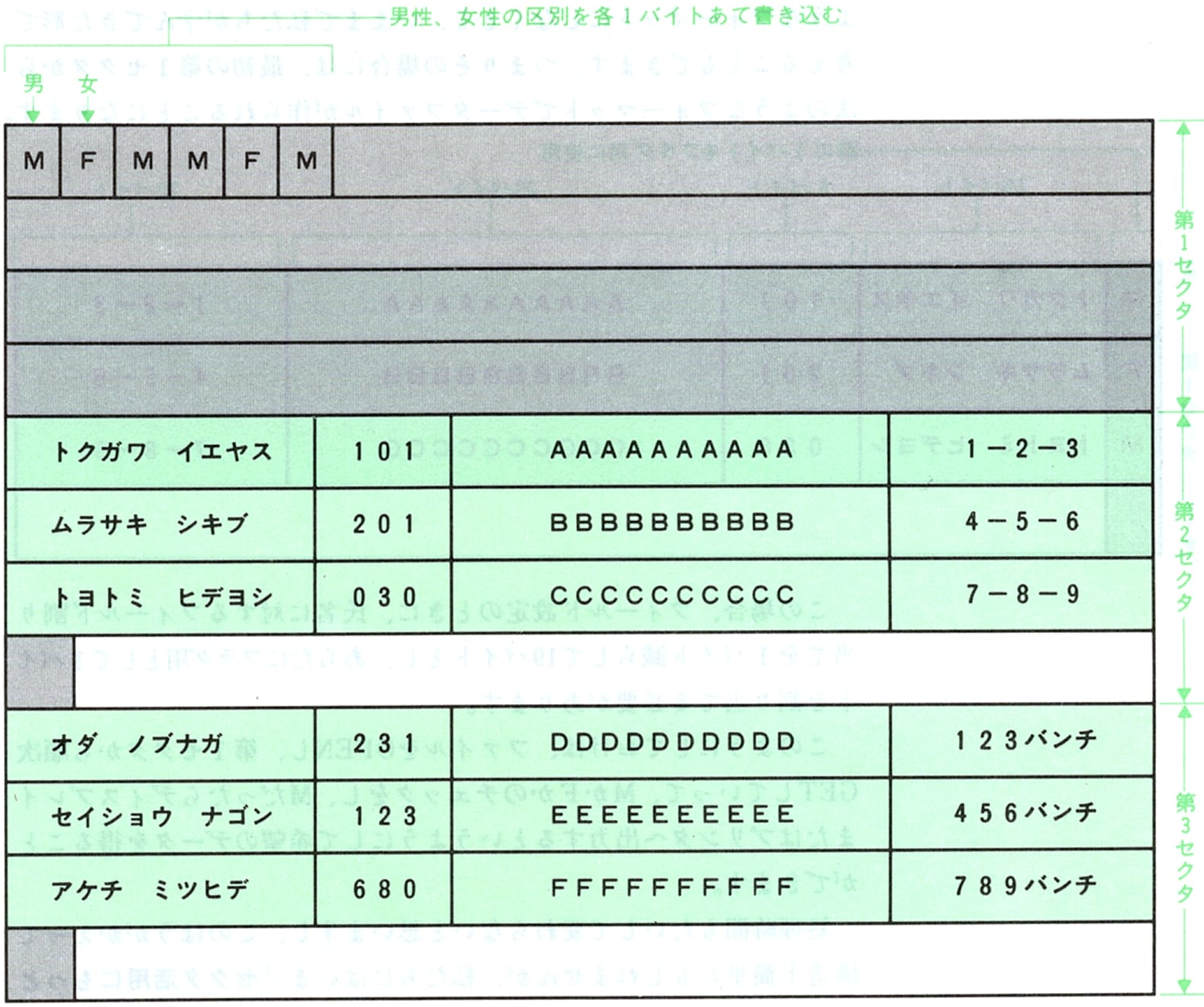
〈注〉この章でのデータの取り出し方が必ずしも合理的というわけではなく、あくまでセクタの使い方に重点があります。

フォーマット

●フォーマットとは形とか様式とかいった意味で、ここではディスクの各セクタの使われ方を指しています。

まあ、こんな話はさておき、このデータファイルのフォーマットを右ページ上のようにしておきますと、1セクタは256バイトですから、第1セクタには256人分の男女のフラグを書き込むことができます。

そして第2セクタ以降から、そのフラグに対応したデータが書き込まれています。つまり、このフォーマットでは256人分のデータしかストックすることができません。ですから、もし、もっとたくさんのデータ、たとえば1000人のデータを扱いたいというような場合には、「M」とか「F」を書き込むセクタをふやせばよいことになります。1000人のデータファイルを作るのでしたら、1000÷256≒3.9ですから、4



第4セクタ以降も第2、第3セクタと同じフォーマットで続く

セクタをフラグ書き込みように確保しておけばよいことになります。

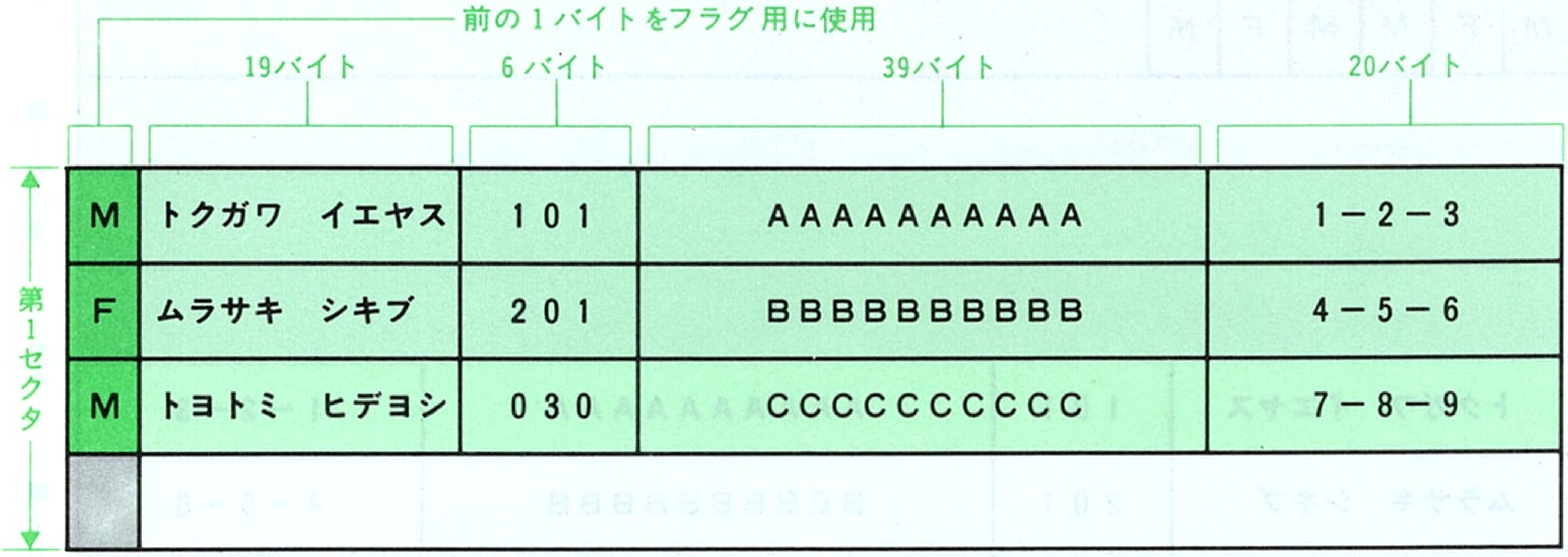
この場合、住所、氏名などのデータは第5セクタから書き込むことになります。

このように、ディスクの容量が許す範囲内で、どのようにも設計することができるのです。しかし、一度フォーマットを決めてしまうと、それを変更するには、古いデータファイルを新しいデータファイルに作りなおすプログラムを作ったりしなければなりませんので、かなりめんどうです。

ということは、**何年かはフォーマットを変更しないですむように、ある程度長期的な視野に立って考えたほうがよい**と思います。

さて、ここで1つだけ補足しておきますが、男性のデータだけを取り出したり女性のデータだけを取り出したりする場合、別に上の図の

ようなフォーマットにしなくとも、いままで私たちが学んできた形で考えることもできます。つまりその場合には、最初の第1セクタから次のようなフォーマットでデータファイルが作られることになります。



この場合、フィールド設定のときに、氏名に対するフィールド割り当てを1バイト減らして19バイトとし、あらたにフラグ用として1バイトを割り当てる必要があります。

このようにしておけば、ファイルをOPENし、第1セクタから順次GETしていって、MかFかのチェックをし、Mだったらディスプレイまたはプリンタへ出力するというようにして希望のデータを得ることができます。

処理時間もたいして変わらないと思いますし、このほうがかえって構造上簡単かもしれませんが、私たちにはいま**「セクタ活用にもっと汎用性をもたせる」**というテーマが与えられていますので、あえて、こちらは採用しないことにします。

また、この章では、セクタの新しい使い方を覚えていただくことが目的ですので、プログラムも簡単にし、この章で得た知識をもとに、次章でもう少し実用的なプログラム作りをこころみようと思います。

次のステップへの予備知識といったところですので、扱えるデータ数も256人までとします。

1・2　データの書き込み

それではまず、データ書き込み部分のプログラムから作ることにしますが、最初は初回のデータ書き込みだけを扱い、データを追加する場合は、このプログラムが完成したら、それを手直しして作っていくことにします。

プログラミングをする場合、最初からいっきに作り上げてしまおうとしないで、**ある柱となるプログラムを作り、それに枝葉をつけていくというステップを踏んだほうが間違いも少ないですし、作りやすい**ようです。

前編の読者の方へ

この本の前編をお読みになった方もかなりいらっしゃると思いますので、少しだけ付け加えておきます。

この本では、前編から連続して入れるように、テーマもあまりへだたりのないものを選択しました。また、プログラム中で使用する変数名も、できるだけ前編に近い形で使うことにしました。この点、みなさんが理解しやすいように便宜をはかったつもりです。

ところで本書では、前回と同様、プログラムファイルはドライブ1側のディスクにセーブし、データファイルはドライブ2側のディスクに作ることを原則としました。　**使用するディスク**

未使用のディスクを2枚用意し、N88BASICを起動してから、ユーティリティプログラム「dskut2.j88」を実行し、メニューの中から「**システムディスクの作成**」を選び1枚目のディスクをシステムディスクにします。次に同じくユーティリティプログラムにより2枚目のディスクからデータディスクを作ります。このようにして作成したシステムディスクをドライブ1に、データディスクをドライブ2にセットして第3章まで読み進んでください。　**ディスクの割り当て**

プログラミングの途中で、バグがでたり、こんなことをつけ加えたらもっとよいのでは、といったこともでてくると思いますが、そんなときはみなさんと一緒に考え、修正していきたいと思います。

それからもう1つ、プログラムの中で1行のステートメントが長くなるものは、途中で折り返してありますので、このことを前提にキーインしていってください。

初期設定

それではまず初期設定の部分から。

```
100 ' file name ････> dmail1.bas
110 ' 1985.2.10
120 ' ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｶｷｺﾐ
130 ' save "dmail1.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
```

これはもうおなじみのステートメントばかりですね。

行番号110の年月日は、プログラム作成日でしたね。

行番号130のステートメントの意味はわかりますね。そうです。このプログラムをセーブするときに使うのでした。わからない人は、前の本をもう一度ひっくり返してください。

画面モードの指定

このプログラムでの画面モードは前回同様**80桁×20行**です。

また、ファンクションキー表示はじゃまですので消しておくことにしましょう。**CONSOLE文の第3パラメータを0**にします。テキスト画面はモノクロにしますので**第4パラメータも0**です。

変数の整数型宣言

行番号160で、A～Zで始まるすべての変数の整数宣言をしています。

配列宣言

行番号170、180は配列宣言です。NM＄が氏名用の、PN＄が郵便番号用の、AD＄は住所用、そしてBA＄が番地用の配列変数です。

SEX＄は何のための配列変数だかわかりますか？ SEXなどというと日本人には少々おかしな響きをもって聞こえがちですが、まともに考えてください。

前節で、第1セクタに男女を区別するフラグを書き込むとお話ししましたが、その**フラグのための配列**です。

男性でしたら「M」を、女性でしたら「F」を書き込むわけですが、いずれにしても1文字書き込めばよいので1名について1バイトあれば十分です。そして1セクタは256バイトですから、第1セクタをまるまるフラグ用として使うとすれば、256人分書き込めます。そして、256個のフラグ用のフィールド設定は、配列を使って行ないますので、256の大きさの配列を宣言しておけばよいことになります。

ファイル内での書き込まれ方

さあ、初期設定ができましたので、使用するファイルのOPENですが、その前に、男女の別、名前、住所などがランダムファイルの中でどのように書き込まれるか図示しておきましょう。

ここでのランダムファイルのディスクリプタは、

2：mail01.dat

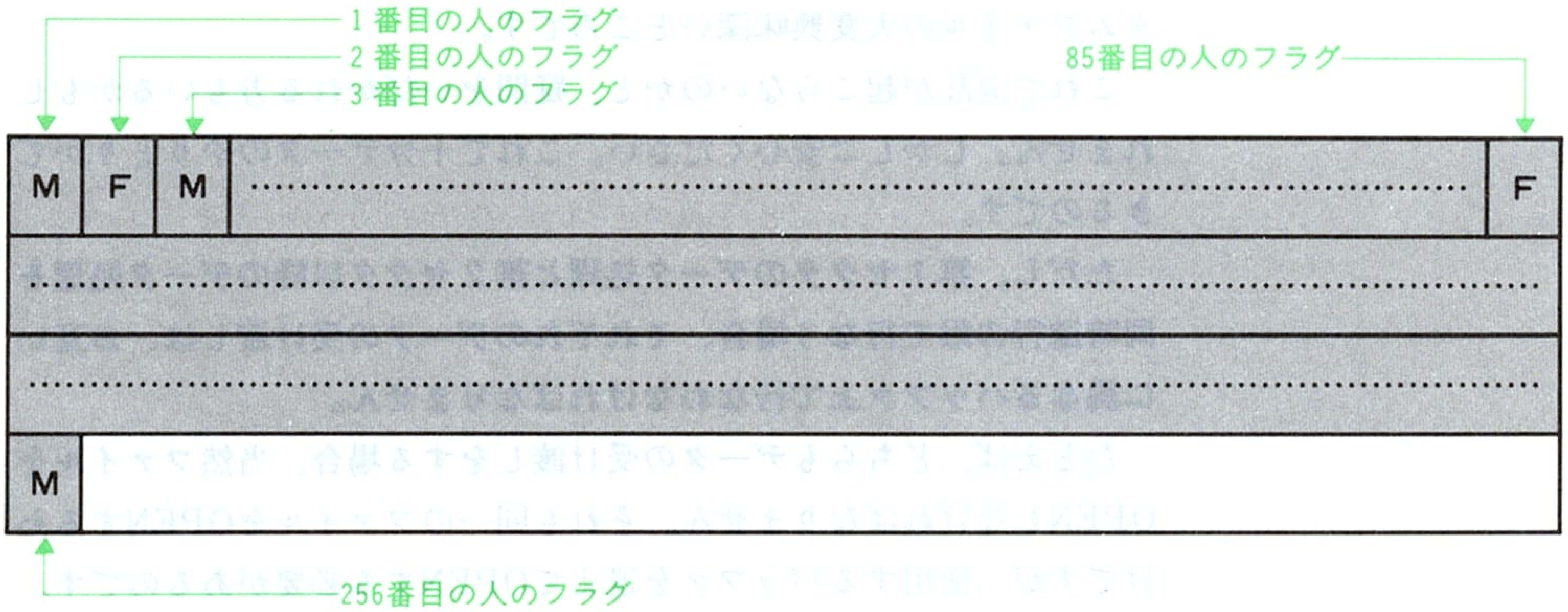
とします。

また、コード用のシーケンシャルファイルについては、

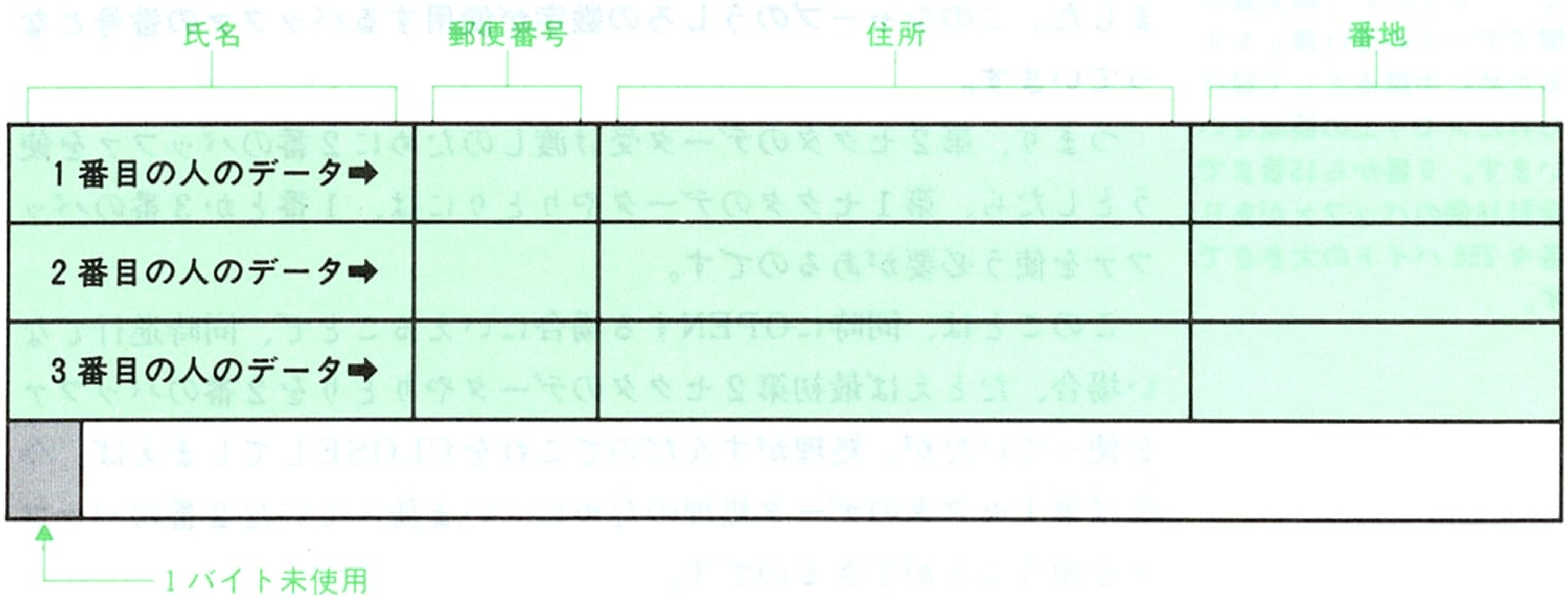
2：indx01.dat

とします。

下の図をごらんください。これは、ランダムファイルの第1セクタ、



⬆ランダムファイル「mail01.dat」内でのデータ格納状況(第1セクタ)



⬆ランダムファイル「mail01.dat」内でのデータ格納状況(第2セクタ)

第２セクタでのデータの格納状況を図示したものです。

第１セクタは、256人分の性別のフラグ用として使用し、それに対応したデータは第２セクタ以降に書き込まれます。

バッファ上でのフィールド割り当て

前ページの図を見て、だいたいの様子をつかんでいただけたと思います。

前編では、第１セクタも第２セクタと同じ**フォーマット**で書き込まれていましたね。ところが今度はまるきり違うフォーマットになっています。

といって、ファイル名が異なるわけではなく、どちらもまったく同じファイル名「mail01.dat」です。つまり、**同一ファイル内に、まったく異なるフォーマットのデータが同居している**のです。これがランタムファイルの大変興味深いところです。

これで混乱が起こらないのかと、疑問をいだかれる方もいるかもしれません。しかしご安心ください。これで十分データのやりとりができるのです。

ただし、第１セクタのデータ処理と第２セクタ以降のデータ処理を同時進行の形で行なう場合、それぞれのデータの受け渡しは、お互いに異なるバッファ上で行なわなければなりません。

たとえば、どちらもデータの受け渡しをする場合、当然ファイルをOPENしなければなりません。それも同一のファイルをOPENするわけですが、使用するバッファを違えてOPENする必要があるのです。

バッファというのはわかりますね？

ファイルをOPENする書式で「＃１」とか「＃２」というのをつけていました。このシャープのうしろの数字が使用するバッファの番号となっています。

つまり、第２セクタのデータ受け渡しのために２番のバッファを使うとしたら、第１セクタのデータやりとりには、１番とか３番のバッファを使う必要があるのです。

このことは、同時にOPENする場合にいえることで、同時進行でない場合、たとえば最初第２セクタのデータやりとりを２番のバッファを使っていたが、処理がすんだのでこれをCLOSEしてしまえば、今度は第１セクタのデータ処理のために、いま使っていた２番のバッファを使うことができるのです。

ちょっとややこしいかもしれませんが、要するに、同一のファイルを異なるフォーマットで同時にOPENしてフィールドを設定する場合、

フォーマット

●「フォーマット」とは型あるいは型式といったくらいの意味で、この場合ディスクに書き込まれるときの様式と表現すればだいぶ近くなります。

バッファ

〈注〉コンピュータのメモリとディスクという磁気盤の間でデータの受け渡しをするために中継点として設けられたメモリ上の領域をいいます。０番から15番まで、合計16個のバッファがあり、各々256バイトの大きさです。

それぞれデータの受け渡しには異なるバッファを使用し、あるバッファを一度CLOSEしてしまえば、そのバッファはまったく開放され、どんなファイルにも使用できるのです。

この辺は、理屈よりも実践をしていったほうが理解しやすいかもしれません。説明はこれまでにして、先へ進むことにしましょう。

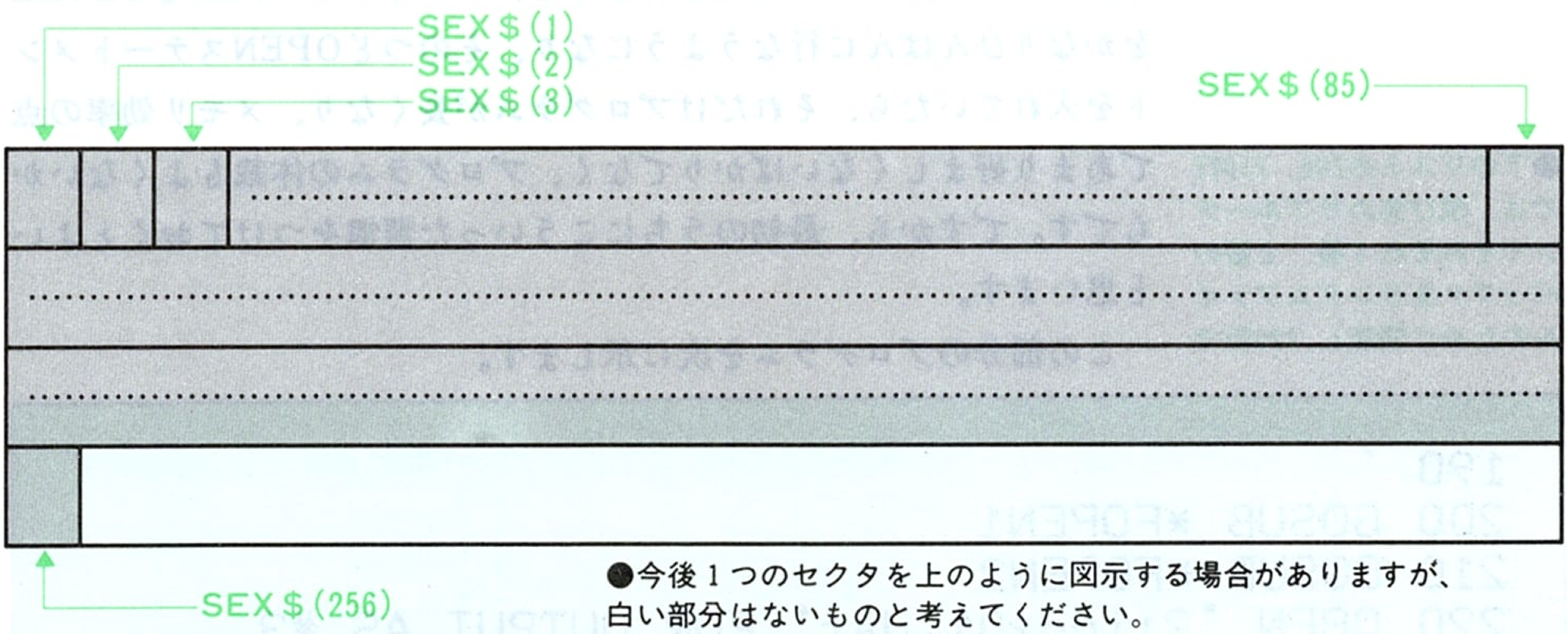
フィールド割り当て

さて、**バッファ上でのフィールド割り当て**ですが、これも図に示すことにしましょう。

第1セクタのデータ処理には2番のバッファを、第2セクタ以降のデータ処理には1番のバッファを使うことにします。

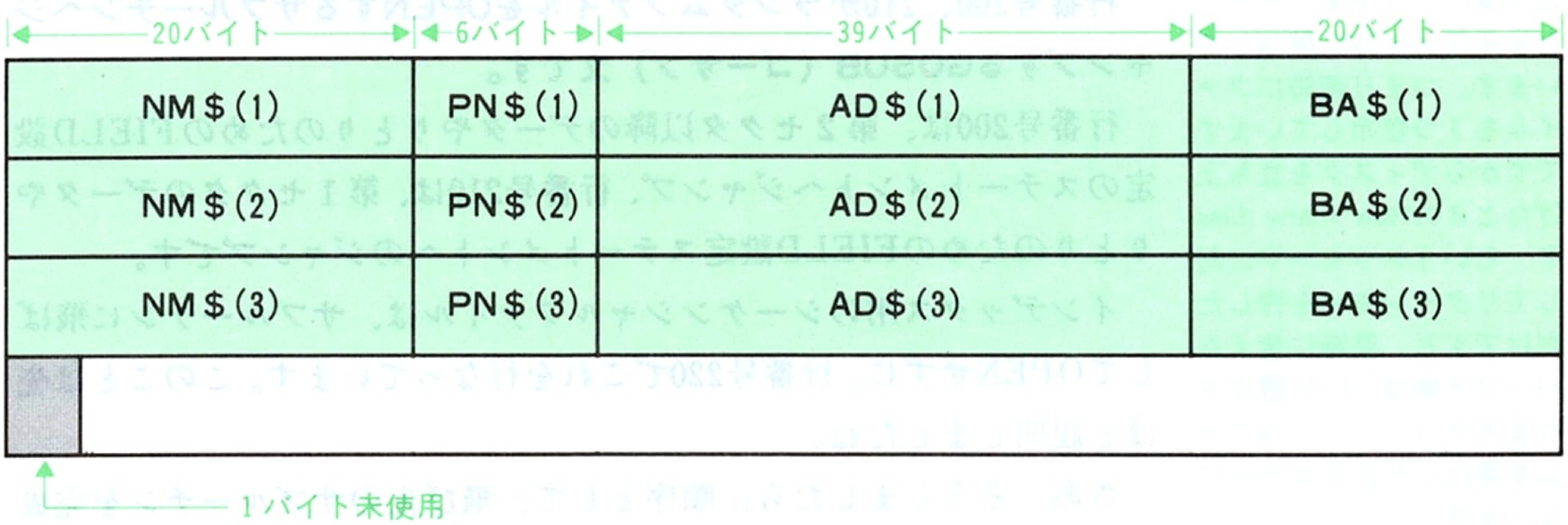
セクタ番号とバッファ番号を対応させればよかったのですが、ご容赦ください。

次の図は、各変数のバッファ上でのフィールド割り当てを示したものです。



●今後1つのセクタを上のように図示する場合がありますが、白い部分はないものと考えてください。

⬆2番バッファでのフィールド設定



⬆1番バッファでのフィールド設定

このプログラムでのフィールド設定、いかがですか？　なんとなく理解していただけたのではないかと思います。

それでは、いよいよファイルをOPENし、**FIELD（フィールド）文**を作っておくことにしましょう。

ファイルをOPENする

ファイルのオープン

このプログラムでは、**ランダムファイルのOPENはサブルーチン化しておく**ことにします。ただし、シーケンシャルファイルはサブルーチンとせず、使用するたびにOPENステートメントを入れることにします。

サブルーチン化するのはなぜかといいますと、このプログラムではあまり意味をもちませんが、あなたがたがこれから徐々に複雑なプログラムを組んでいくようになりますと、ファイルのOPENとCLOSEをかなりひんぱんに行なうようになり、そのつどOPENステートメントを入れていたら、それだけプログラムが長くなり、メモリ効率の点であまり好ましくないばかりでなく、プログラムの体裁もよくないからです。ですから、最初のうちにこういった習慣をつけておくとよいと思います。

この部分のプログラムを次に示します。

●下のリストの200、210行では、飛び先のサブルーチンでそれぞれ１番、２番のバッファをランダムファイルのために開放し、220行では３番バッファをシーケンシャルファイルに開放しています。つまり同時にファイルを３つ使用しています。ですからディスクを立ち上げたときのHow many files？　というメッセージに対してリターンキーを押しただけですと、同時に使えるバッファ数は「２」に設定されますから、このプログラムを実行したときエラーになります。

```
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
220 OPEN "2:indx01.dat" FOR OUTPUT AS #3
230 '
```

行番号200、210がランダムファイルをOPENするサブルーチンへジャンプするGOSUB（ゴーサブ）文です。

行番号200は、第２セクタ以降のデータやりとりのためのFIELD設定のステートメントへジャンプ、行番号210は、第１セクタのデータやりとりのためのFIELD設定ステートメントへのジャンプです。

インデックス用のシーケンシャルファイルは、サブルーチンに飛ばしてOPENせずに、行番号220でこれを行なっています。このことは先ほど説明しましたね。

さあ、そうしましたら、順序として、飛び先のサブルーチンを完成させてしまいましょう。

フィールドの多重定義

まず、順序が逆になりますが、第2セクタ以降のデータの出し入れをするフィールドから設定しておきましょう。

ここでは、氏名、郵便番号、住所、番地用の変数のフィールドを、それぞれ3人分について割り当てるわけですが、配列変数を使って簡潔な形に仕上げたいと思います。

この方法については前編に述べましたのですでにおわかりの方もいると思いますが、フィールド割り当ての様子を図示することによって、もう少し補足しておきたいと思います。

最初にプログラムを示します。

```
1000 '――――――――――――――――――――――――――――――――
1010 *FOPEN1
1020 '――――――――――――――――――――――――――――――――
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
```

このプログラムは、前編の形とほぼ同じですので、ごらんになるとわかると思いますが、ちょっとうなずけないという方は、前編をもう一度ひっくり返して、ある程度理解した上で、この記事に戻ってきていただければと思います。

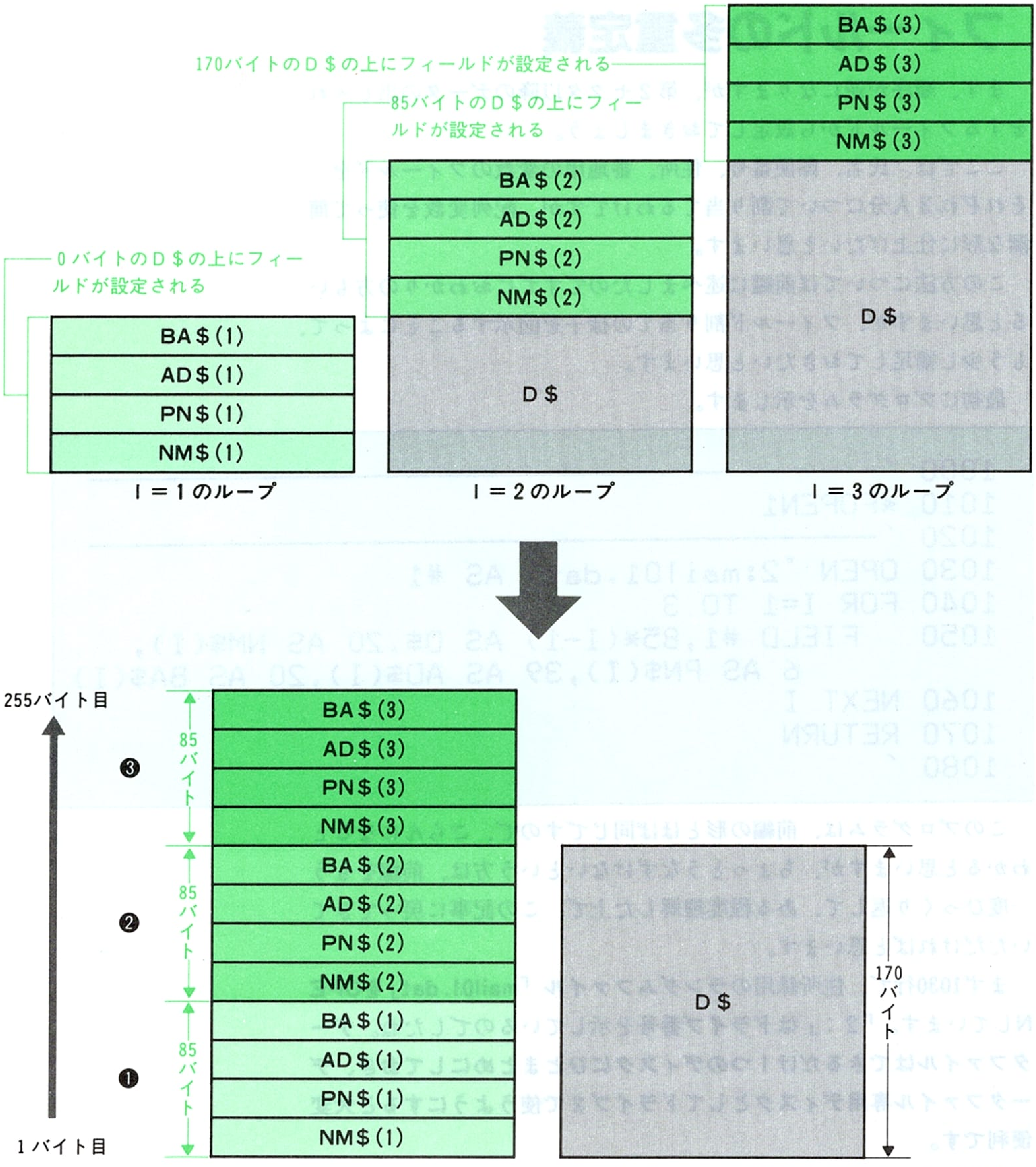
まず1030行で、住所録用の**ランダムファイル「mail01.dat」をOPEN**しています。「**2:**」**はドライブ番号**を示しているのでしたね。**データファイルはできるだけ1つのディスクにひとまとめにしておき、データファイル専用ディスクとしてドライブ2で使うようにすると大変便利です。**

1040行から1060行までがフィールド設定のステートメントです。

1040行のFOR文と1050行のNEXT文ではさみ、3回ループさせ、1回ループするたびに前のフィールドに重ねて定義しています。

この辺のプロセスが初めのうちはわかりにくいと思いますので、この様子を次ページに図示しました。

図を見ながら説明をお読みください。



●FOR〜NEXT文で3回のループが終了すると、フィールド上でデータの受け渡しをできる変数のフィールドは、最終的には下側の図のようになります。それぞれの変数になわ張りが決められたわけです。このほかにもプログラム中ではいろいろな変数が使われますが、バッファ上でデータのやりとりに参入できるのは、上の図の変数のみということになります。D＄もダミーとはいえ、元気に生きているわけですが、私たちがGETさえしなければ混乱は起こらないことにないことになります。

まず**最初のループ**ではI＝1ですから、**D＄**に対するフィールドは

85＊(I−1)=85＊(1−1)=0

となります。0ということ、つまり**0バイトがD＄のフィールド幅として割り当てられます**。ということは、何もないということと同じですね。

〈注〉D＄の「D」はダミー用の変数で「DUMMY」の最初の1文字を使いました。XでもYでもかまいません。

そして、この何もないところに配列NM＄(1)、PN＄(1)、AD＄(1)、BA＄(1)のフィールド幅が20バイト、6バイト、39バイト、20バイトというように続けて設定されていきます。ですからこの段階では、D＄という変数は存在するのですが、フィールド幅は0ということになります。

ここで1つおことわりしておきますが、左ページの図ではD＄以外の変数はどれも同じバイト数になっていますが、実際には異なるバイト数ですのでご了承ください。また、フィールドが設定される様子も、理解しやすいのでこのようにしました。

次は**2回目のループ**、I＝2の場合ですが、今度はD＄に対するフィールドは、

85＊(I−1)=85＊(2−1)=85＊1=85

となります。

つまり、**85バイトがD＄のフィールド幅として割り当てられ、その上にNM$(2)などのフィールドが設定されます。**

ここでよく誤解しやすいのですが、D＄に85バイトが割り当てられたので、I＝1のループで設定されたNM＄(1)〜BA＄(1)のフィールド85バイトはなくなってしまうのではないかという心配がでてきますが、これはまったく支障ありません。

というのは、上のような形で一度フィールド幅が割り当てられれば、一種の権利が生じたようなもので、消えることはありません。

最後に**3回目のループ**、今度はI＝3ですから、D＄に対するフィールド幅は、

85＊(I−1)=85＊(3−1)=85＊2=170

と、だいぶふくれ上がりました。2回目のループのときのD＄の中身85は追い出されて、今度は170がその内容となりました。そして、**この170バイトの上に、NM＄(3)〜BA＄(3)のフィールドが割り当てられるのです。**

これで、3回のループが終了し、フィールド設定ができたわけですが、最終的なテリトリーの構図はどうなったでしょうか。これを示すのが20ページの下の図です。

ファイルをOPENしたときバッファ上で活躍できる変数名およびそのテリトリーは図に示す通りです。

たとえばPN＄(1)でしたら、1セクタ内の21バイト目から6バイト、つまり25バイト目までが管轄になるわけです。あくまで管轄ですから、**GET（ゲット）文、PUT（プット）文**でお呼びがかからなければ活躍のしようがありません。

一方、このプログラムでの処理に関係のないD＄も図でわかるようにテリトリーを与えられています。このなわ張りは大きいですね。❶と❷を合わせた**170バイト**です。ですから、**GET文でセクタをGETし、D＄を呼び出せば、1バイト目から170バイト目までのデータがD＄に入ってしまいます。**

たとえば、❶と❷の位置に氏名と住所などが書き込んであれば、2人分のデータをまるまるのみ込んでしまうことになります。このことは逆にPRINT文を使ってD＄の内容をディスプレイに表示させれば、2人分のデータをいっぺんに表示させることができる、ともいうことができます。

ストリング

〈注〉バッファを介してデータの受け渡しをするときは必ず、ある一定のフィールド幅を割り当てられた文字型変数を通して行われます。

この場合にはどのデータも**文字列（ストリング）**ですからよいのですが、これが数値データなどですと、メチャクチャな値になってしまいます。

では次に、第1セクタのデータやりとりのための変数のフィールド割り当てです。

これもやはり、**FOR～NEXT（フォア～ネクスト）文**を使って、きれいに設定したいと思います。

行番号を2000からとして次のようにキーインします。

```
2000 '―――――――――――――――――――――――
2010 *FOPEN2
2020 '―――――――――――――――――――――――
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
```

2030行はランダムファイルのOPENです。

使用するバッファは２番を指定しています。1030行でOPENしたファイル名と同じですから、もしここで１番バッファを指定してRUNすると、次のようなエラーメッセージが現れてプログラムがストップします。

```
File already open in 2030
Ok
```

2020行で、すでにOPENしてあるファイルをOPENしようとした、というメッセージです。

プログラム実行中にこのようなエラーがでたら、あわてずに、**まずダイレクトモードで「CLOSE ⏎」をキーインし、**バッファ番号をなおしたりするとよいでしょう。

〈注〉エラーメッセージなどが出てプログラムがストップしたときは必ずダイレクトモードでCLOSEをキーインしておきましょう。つまり、開いたファイルを閉じておくのです。

ここで１つ注意していただきたいことは、第１セクタのデータやりとりは常に２番バッファを使うと決められているわけではないことです。1030行でOPENしたバッファがCLOSEされていれば、１番バッファを使ってもかまいませんし、３番バッファがあいていれば、このバッファでもかまいません。要は、**同時に２つのファイルで同一バッファを使うことができない**ということです。

1030行と2030行でOPENするファイル名は同じですが、それぞれフィールド設定が異なるので、同じバッファを同時に使うことができないのです。

2040行から2060行のFOR～NEXTループで、性別を見分けるフラグ用変数のフィールドを設定しています。

これはもうわかりますね。256回ループさせて、SEX＄(１)からSEX＄(256)までの配列にそれぞれ１バイトずつフィールドを割り当てています。

この256回のループの過程で、ダミー用変数Ｄ＄のフィールドは０から始まり255まで変化し、**最終的には255が**残ります。

〈注〉先ほど1000行からのサブルーチンでダミーＤ＄にはバッファ１番上で170バイトの領域が与えられていましたが、2000行のサブルーチンでダミーとしてＤ＄を使っていますので、こちらが優先され、バッファ２番上でのみＤ＄は有効になります。

この部分のフィールド設定では、256バイトまるまる使っています。未使用バイト０ですから、すごく効率よく使うことができるわけです。

以上、ファイルのOPENとフィールド設定、ご理解いただけましたか？　わかりやすいようにと気を配ったつもりですが、もし行き届か

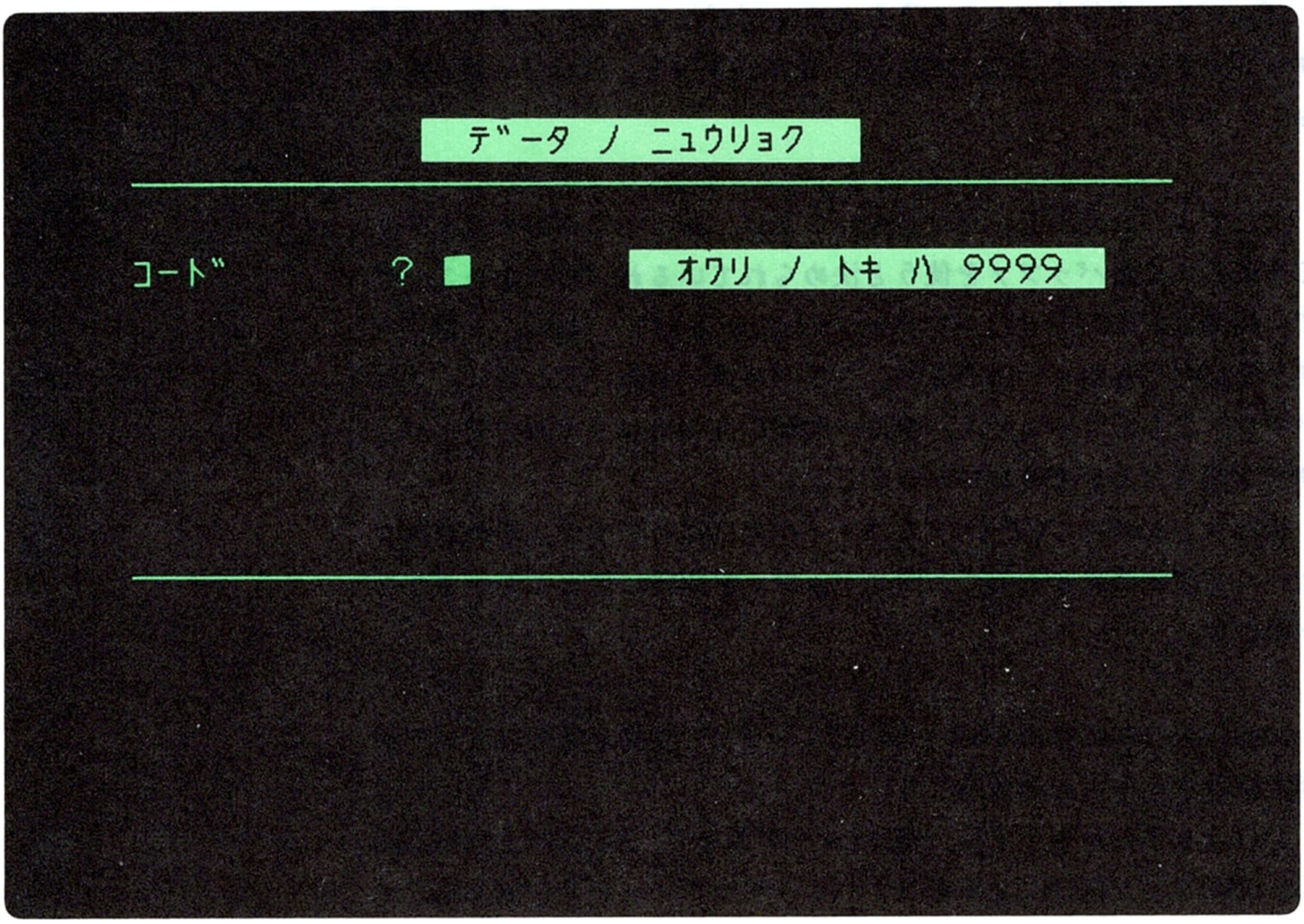
ない点がありましたら、もう一度最初から読みなおしてください。あるいは前著を参考にされるのもよいかと思います。

入力画面について

データの入力方法は前編とだいたい同じです。

入力画面のイメージをどうするかということが問題になりますが、まず、このプログラムを実行すると次のような初期画面が出てくるようにします。

入力画面のポイントは、自分一人ではなく、他の人が見てもわかるようにしておくことです。そのためには、わかりやすいメッセージ表示も大切な問題になります。



〈注〉上の入力画面は実際の画面とは多少ズレがありますがイメージ的にはほぼ同じものです。

上下にグラフィックキャラクタを使って横線が入り、上に、このプログラムの処理内容が**白黒反転して表示**されます。

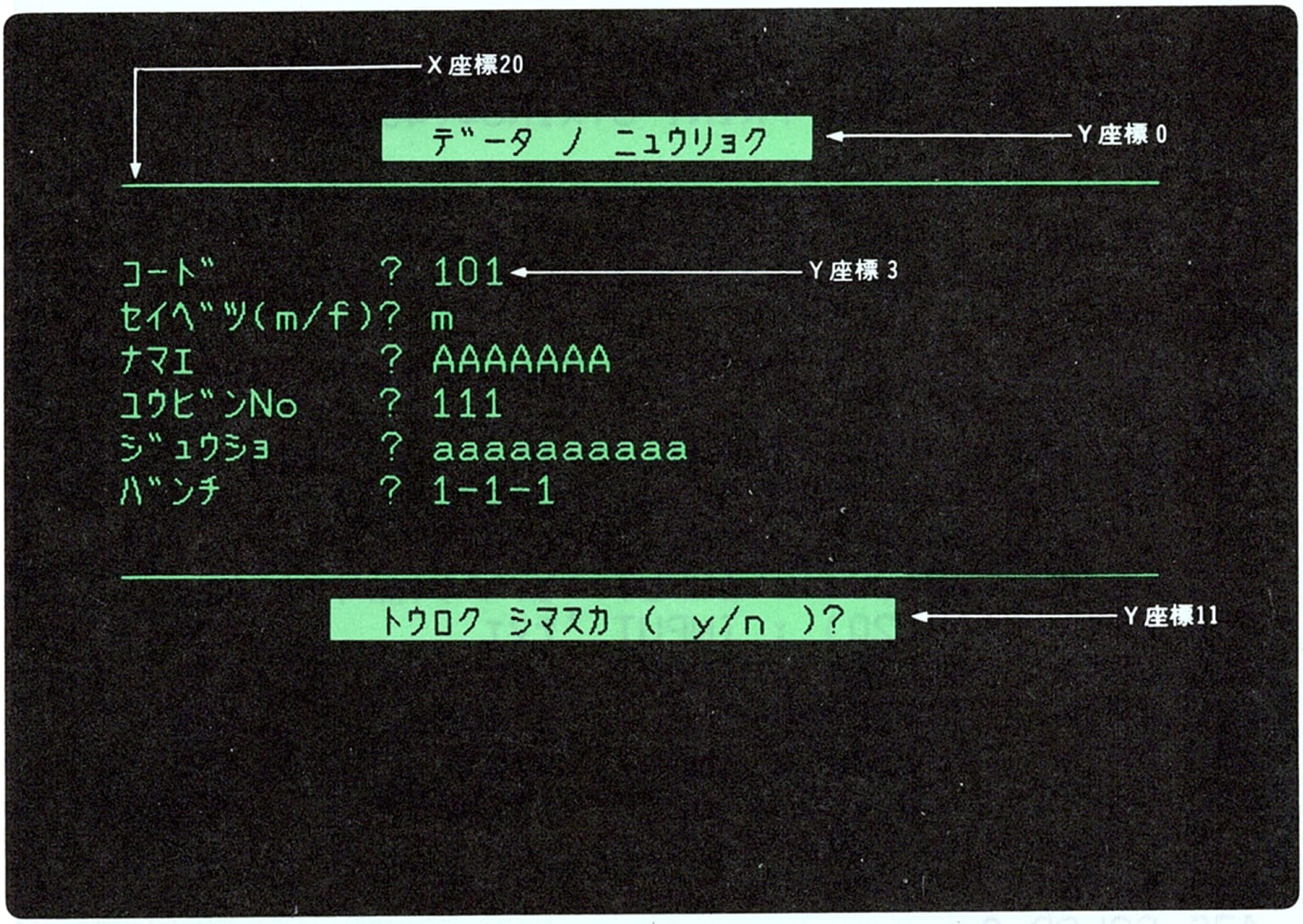
上下の罫線の間がデータ入力行になりますが、最初に**コード入力のプロンプト文**が表示され、そのとなりに

オワリ　ノ　トキ　ハ　9999

というメッセージが反転ブリンクしています。つまりここで「**9999**」をキーインすれば、データ入力モードから抜け出し、次の処理へと進

みます。

次に示すのは６つの項目を入力し終え、登録するかどうかの確認をうながしているところです。



ここで「Y」を入力すればディスクへデータが書き込まれ、「N」を入力すればもう一度最初の項目からやりなおしです。

〈注〉確認メッセージに対しては大文字でも小文字でも機能するようにしてあります。

データ入力部分のプログラム

この部分は、みなさんはすっかりマスターされていると思います。次ページのようにキーインすればよいですね。

モノクロモードのときの**COLOR文**のパラメータに６を指定すると反転表示され、文字は**ブリンク（点滅）**します。330～340行がこれに相当します。つまり、コード入力に入る前に、

オワリ　ノ　トキ　ハ　9999

のメッセージが反転ブリンクで出力されます。画面では

コード　　？

のプロンプト文の右側に表示されますが、プログラムでは先に出力しておかなければなりません。

COLOR文

●モノクロモードでパラメータに４を指定すると反転表示、６を指定すると反転してブリンクします。なお反転・ブリンクの操作はカラーモードでは使えませんので注意してください。

```
240 COLOR 4
250 LOCATE 31,0 : PRINT "  ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ"
260 COLOR 0
270 LOCATE 20,1 : PRINT STRING$(40,149)
280 LOCATE 20,10: PRINT STRING$(40,149)
290 '
300 '―――――――――――――――――――――――――――――――――――
310 *LOOP
320 '―――――――――――――――――――――――――――――――――――
330 COLOR 6
340 LOCATE 40,3 : PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  "
350 COLOR 0
360 LOCATE 20,3 : INPUT "ｺｰﾄﾞ         ";CD
370 IF CD=9999 THEN GOTO *OWARI
380 LOCATE 40,3 : PRINT SPC(20)
390 PRINT TAB(20) : INPUT "ｾｲﾍﾞﾂ(m/f)";A1$
400 PRINT TAB(20) : INPUT "ﾅﾏｴ       ";A2$
410 PRINT TAB(20) : INPUT "ﾕｳﾋﾞﾝNo   ";A3$
420 PRINT TAB(20) : INPUT "ｼﾞｭｳｼｮ    ";A4$
430 PRINT TAB(20) : INPUT "ﾊﾞﾝﾁ      ";A5$
440 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
450 COLOR 4 : LOCATE 27,11
460 PRINT "   ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?    "
470 COLOR 0
480 I$=INKEY$ : IF I$="" THEN GOTO 480
490 IF I$="N" OR I$="n" THEN GOTO *KESU
500 IF I$="Y" OR I$="y" THEN GOTO *KAKU
510 BEEP : BEEP : GOTO 480
520 '
```

360行でコード入力が終れば

オワリ　ノ　トキ　ハ　9999

SPC関数

のメッセージは必要なくなりますので、380行で**SPC(スペース)関数**によりスペースを出力して表示を消しています。

360行でコードを入力し、もし「**9999**」がキーインされたら、現在オープンしているファイルをすべてクローズし、プログラムの処理を終りにします。

この判定は370行のIF文で行っています。

390行から430行までが登録する人のデータ入力です。

480行は**INKEY＄文**によるキー入力待ちですね。何かキー入力があ

るまで、この480行から抜け出すことができません。

INKEY＄にかかわる問題として、よく読者の方から問い合わせがありますが、その主旨はたいてい正常に動作しないということです。

よく聞いてみますと、多くの場合次の２点の誤りに原因があるようです。

❶ I$＝″″の２つのダブルコーテーションの間にスペースを１つ入れてしまった。

❷ キー入力待ちに対して大文字あるいは小文字のＹまたはＮを入力しているのだが、うまく動作しない。この場合には、それ以前のデータ入力で カナ キーをロックしてあるために、Y または N キーを押しているにもかかわらず、カナが入力されてしまったためです。

気がついてみれば、なあんだという問題ですが、自分でこの間違いをおかしているときには、なかなかわからないものです。

横道にそれたついでにもう１つ。

それは、何かのつごうでファイルをオープンしたままプログラムをストップさせたりしたときに、必ずCLOSEをキーインしてくださいと説明してきましたが、これはCLOSEでなく**END**を使っても同じことです。

END文を実行すると、現在オープンしているファイルをすべて自動的にクローズしてくれます。

どちらでもお好きなほうをどうぞ……。

さて、それでは本題に戻りましょう。

460行の確認メッセージに対して490行で「N」または「n」が入力された場合には、もう一度入力しなおさなければなりません。修正のステートメントを追加して、すべてのデータをキーインしないですむようにもできますが、ここでは省略します。みなさんが、あとでチャレンジしてみてください。

OKであれば500行でデータをディスクに書き込むルーチンに飛ばします。510行は、とんでもないキーを押してしまったときの処理ですね。

ディスクへのデータ書き込み

さあ、データ入力部分ができました。次は、入力データの書き込みです。

書き込みは、**コードはシーケンシャルファイルですからWRITE文で、また性別、氏名、郵便番号、住所、番地はランダムファイルですからLSET文**を使えばよかったですね。

ただ、前著と違うところは、ランダムファイルにおいて、性別を示すフラグは第1セクタに、氏名その他は第2セクタ以降にというように、それぞれ異なるセクタに同時に書き込まなければならないという点です。

ですから、**書き込みの前にまず、書き込むべき位置を決めておかなければなりません。**

では、プログラムをキーインする前に、この問題を解決しておきましょう。

最初に、性別を識別するためのフラグですが、これは第1セクタに入力された順に書き込んでいけばよいわけですね。ですから、**入力順を表わす変数をPとでもして、1つずつカウントしていけばよいのです。**

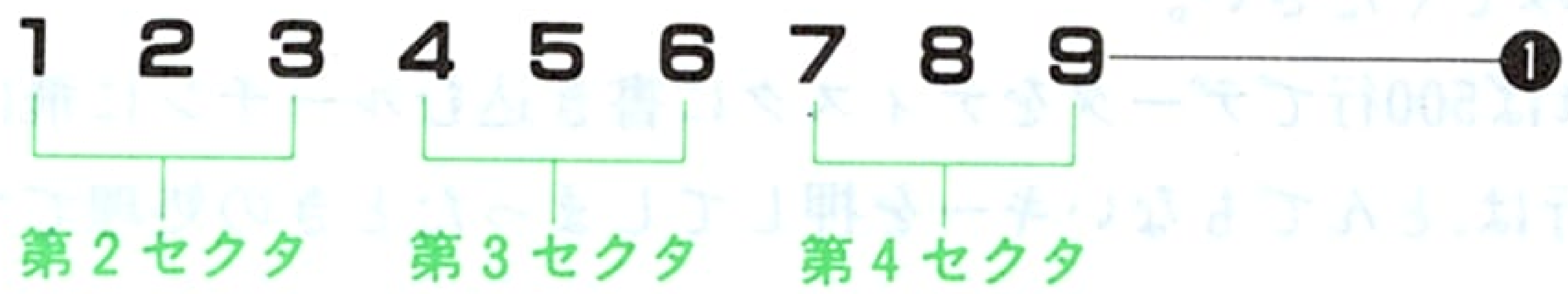
いいですか？　この変数をカウンタ代わりにしていますから、1番目の人から順に、1、2、3……とカウントされ、このときの値が変数Pに入ります。

つまり、3人目の人のときは、この変数の中身は3になりますから、SEX＄(3)にA1＄の内容をバッファ上で**LSET（エルセット）**すればよいのです。

性別用のフラグの書き込みはわかりましたね。

次は第2セクタ以降です。いいですか。

第2セクタ以降には、1セクタ内に3人分のデータが書き込まれます。そして、第2セクタの頭から、1番目の人、2番目の人、3番目の人のデータと順に書き込まれ、4番目の人からは第3セクタに移ります。これを図式化すると次のようになります。**黒くて太い数字が1番目の人、2番目の人……を表しています。**



私たちが欲していることは、上の「1　2　3」のようにそれぞれ異なる数字を、セクタを表す共通の数「2」に換算できることです。

では、❶の数字を3で割って、その商を求めるとどうなるでしょうか。次のようになりますね。

0 0 1 1 1 2 2 2 3

3つ1組の数列のうちで1つだけが仲間はずれになっています。たとえば、「0　0　1」の組の「1」のように……。

これを3つとも同じ商にすることができれば、その商に定数を加えてセクタ番号を算出することができます。

もう、どうすればよいかわかりますね。仲間はずれといっても1だけずれているわけですから、❶の数列からあらかじめ1だけ引いてずらしておき、それを3で割って商を求めればよいわけですね。

やってみましょう。

❶の数列からそれぞれ1を引くと

0 1 2 3 4 5 6 7 8

となります。

では、この数列を3で割って商を並べてみますと、

0 0 0 1 1 1 2 2 2

となりますね。どうですか？　きれいにそろったでしょう。

それではこの数列に定数「2」を加えてみましょう。

2 2 2 3 3 3 4 4 4

第2セクタ　第3セクタ　第4セクタ

定数を加算したら、それぞれの数字が、それぞれ書き込まれるべきセクタ番号になりました。やっと私たちの念願がかなったわけですね。ここまでくれば、あとは楽です。

❶の数列は、入力される順番ですから、変数Pにしまわれています。ですから、この変数を使えば、**書き込まれるセクタ番号**は次のように表わすことができます。

書き込みセクタの算出

SEC=(P−1)¥3+2

どうやら峠を越したようですが、もう1つ問題が残っています。それは、書き込まれるセクタはわかりましたが、そのセクタ内で何番目

の位置に書き込んだらよいか、ということです。

これも、いまと同じように、28ページの❶の数列をもとに換算する式を作ってみましょう。

❶の数列で表わされる人たちのデータは、次のように、各セクタ内の緑色の太い数字で示された位置に書き込めばよいのです。

第2セクタ　第3セクタ　第4セクタ

1 2 3 4 5 6 7 8 9 ―― ❷

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

1 2 3 1 2 3 1 2 3 ―― ❸

今度は、**❷の数列を3で割って、その余りを求めてみましょう。**前は商でしたが、今度は剰余ですよ。

1 2 0 1 2 0 1 2 0

上の❸の数列の「3」に相当する部分がすべて「0」になっています。この「0」を「3」にすればよいのですが、3で割り算をしているのですから、余りが3になることはありませんね。

しかたないですから、余りが左から右へ昇順に並ぶように調整します。❷の数列から1を引いて、それを3で割り余りを求めてみてください。次のようになりますね。

0 1 2 0 1 2 0 1 2

どうですか？　昇順に並びましたね。そうしたら、この剰余にそれぞれ1を加えれば、❸で示す数列が得られますから、この方法によって、セクタ内で書き込まれる位置が決定できるわけです。

書き込まれる位置を示す変数をRECとすれば、次のようにして換算することができます。

REC=(P−1) MOD 3+1

さあ、これでプログラムの下準備ができました。

少々、まわりくどい説明でしたが、ご理解いただけましたか？

ランダムファイルを使ってセクタを有効に、かつ便利に活用しようと思うと、どうしても、いま説明した書き込みセクタの算出と、セクタ内で何番目に書き込んだらよいかの位置の算出はまぬがれません。もし、このような方法を使わないとすると、IF文ばかり多くなり、大変非効率的なプログラムになってしまいます。

それでは次に、書き込みの入口部分のプログラムを示します。

```
530 '―――――――――――――――――――――――――――――
540 *KAKU
550 '―――――――――――――――――――――――――――――
560 P=P+1 ←――――――――― カウンタ
570 SEC=(P-1) ¥ 3+2 ←――――――――― 書き込みセクタの算出
580 REC=(P-1) MOD 3+1 ←――――――― セクタ内での書き込み位置
590 IF REC>1 THEN GET #1,SEC : GET #2,1
600 WRITE #3,CD ←――――――――― シーケンシャルファイルの書き込み
```

行番号560から580までが、これまでにご説明した部分です。

説明は長かったのですが、まとめてみるとこんなに短かくなってしまうのですから、ちょっとがっかりしますね。

行番号590はわかりますか？

セクタ内での書き込まれる位置が２番目、３番目のデータの場合は、バッファ上でデータをLSETする前に、そのセクタの内容をバッファ上に取り出しておく必要があります。

このプログラムは初回の書き込みのためのものですから、590行はなくても正常に動作すると思いますが、データ追加などを考慮すると、追加データを、あるセクタの３番目くらいから書き込む場合もありますので、このようなときには**このセクタをGETしておかないと、すでに書き込まれている１番目、２番目のデータが消えてしまいます。すでに書き込まれているデータがあるときには必ずそのセクタをGETしてからPUTするということ**、これはプログラミングの過程でつい忘れられがちですが、注意してください。

600行でコードをシーケンシャルファイルに書き込み、次に示すプログラムで各データをランダムファイルに書き込みます。

LSET
●レフトセットの略でデータを左詰めでセットします。これに対して右詰めでセッとするときはRSET（ライトセット）を使います。

```
610 LSET SEX$(P) =A1$ ┐
620 LSET NM$(REC)=A2$ │
630 LSET PN$(REC)=A3$ ├―― バッファ上でデータをセット
640 LSET AD$(REC)=A4$ │
650 LSET BA$(REC)=A5$ ┘
660 PUT #1,SEC : PUT #2,1 ←――――― ランダムファイルへの書き込み
670 '
```

610行から650行で、**各データをバッファ上でLSET**していますが、Ａ１＄は２番バッファで、Ａ２＄からＡ５＄は１番バッファで、それぞれフィールド設定された変数にセットしています。

そして、このバッファ上でセットされたデータを、660行のPUT(プット)文でディスクに書き込んでいます。

PUT文
●PUTは英語で「何か物を置く」などのとき使う表現ですね。「ディスクの上にデータを置く」くらいに考えてください。

入力行の消去

ディスクへの書き込みが終っても、また460行の登録するかどうかの問いに「N」と答えても、いずれにしても次の**入力行の消去処理**を行い、再びコード入力に戻します。

```
680 '─────────────────────────
690 *KESU
700 '─────────────────────────
710 LOCATE 20,3
720 FOR I=1 TO 6
730   PRINT TAB(20);SPC(40)
740 NEXT I
750 LOCATE 20,11 : PRINT SPC(40)
760 GOTO *LOOP
770 '
```

TAB(タブ)関数
SPC(スペース)関数
〈注〉TABもSPCも同じように空白を出力しますが、TABは左端から、SPCは左端からとは限っていないことに注意してください。

LOCATE文で最初のスペース出力位置は決めておかなければなりません。あとは**TAB関数**、**SPC関数**の組み合わせで、現在画面に表示されているデータを消去すればよいでしょう。

終りの処理

そして最後にコード入力に対して「**9999**」が入力された場合の終りの処理です。次を入力してください。

```
780 '─────────────────────────
790 *OWARI
800 '─────────────────────────
810 CLS
820 CLOSE
830 RUN "menu1.bas"
840 '
```

820行のCLOSE文にはファイル番号の指定がありませんが、省略された場合は**現在OPENしているすべてのファイルをCLOSE**することになります。

830行でやっているように、終りのときも、とりあえず**メニューへ戻る**ようにしておきます。

メニュープログラムはあとで作りましょう。

それでは忘れないうちにプログラムをセーブしておきましょう。

```
save "dmail1.bas" ↵
```

次に、本節でのプログラムリストを示しておきます。

```
100 ' file name ････> dmail1.bas
110 ' 1985.2.10
120 ' ランダ゛ム デ゛ータ ノ カキコミ
130 ' save "dmail1.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
220 OPEN "2:indx01.dat" FOR OUTPUT AS #3
230 '
240 COLOR 4
250 LOCATE 31,0 : PRINT "  デ゛ータ ノ ニュウリョク  "
260 COLOR 0
270 LOCATE 20,1 : PRINT STRING$(40,149)
280 LOCATE 20,10: PRINT STRING$(40,149)
290 '
300 '────────────────────────────────────────
310 *LOOP
320 '────────────────────────────────────────
330 COLOR 6
340 LOCATE 40,3 : PRINT " オワリ ノ トキ ハ 9999  "
350 COLOR 0
360 LOCATE 20,3 : INPUT "コート゛        ";CD
370 IF CD=9999 THEN GOTO *OWARI
380 LOCATE 40,3 : PRINT SPC(20)
390 PRINT TAB(20) : INPUT "セイヘ゛ツ(m/f)";A1$
400 PRINT TAB(20) : INPUT "ナマエ         ";A2$
```

```
410 PRINT TAB(20) : INPUT "ﾕｳﾋﾞﾝNo     ";A3$
420 PRINT TAB(20) : INPUT "ｼﾞｭｳｼｮ     ";A4$
430 PRINT TAB(20) : INPUT "ﾊﾞﾝﾁ       ";A5$
440 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
450 COLOR 4 : LOCATE 27,11
460 PRINT "    ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?      "
470 COLOR 0
480 I$=INKEY$ : IF I$="" THEN GOTO 480
490 IF I$="N" OR I$="n" THEN GOTO *KESU
500 IF I$="Y" OR I$="y" THEN GOTO *KAKU
510 BEEP : BEEP : GOTO 480
520 '
530 '──────────────────────────────────────────
540 *KAKU
550 '──────────────────────────────────────────
560 P=P+1
570 SEC=(P-1) ¥ 3+2
580 REC=(P-1) MOD 3+1
590 IF REC>1 THEN GET #1,SEC : GET #2,1
600 WRITE #3,CD
610 LSET SEX$(P) =A1$
620 LSET NM$(REC)=A2$
630 LSET PN$(REC)=A3$
640 LSET AD$(REC)=A4$
650 LSET BA$(REC)=A5$
660 PUT #1,SEC : PUT #2,1
670 '
680 '──────────────────────────────────────────
690 *KESU
700 '──────────────────────────────────────────
710 LOCATE 20,3
720 FOR I=1 TO 6
730   PRINT TAB(20);SPC(40)
740 NEXT I
750 LOCATE 20,11 : PRINT SPC(40)
760 GOTO *LOOP
770 '
780 '──────────────────────────────────────────
790 *OWARI
800 '──────────────────────────────────────────
810 CLS
820 CLOSE
```

```
830 RUN "menu1.bas"
840 '
1000 '―――――――――――――――――――――――――
1010 *FOPEN1
1020 '―――――――――――――――――――――――――
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
         6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '―――――――――――――――――――――――――
2010 *FOPEN2
2020 '―――――――――――――――――――――――――
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
```

1・3　プログラムの実行

セクタの扱いを変えた初めてのプログラムです。さっそく実行してみたいと思います。

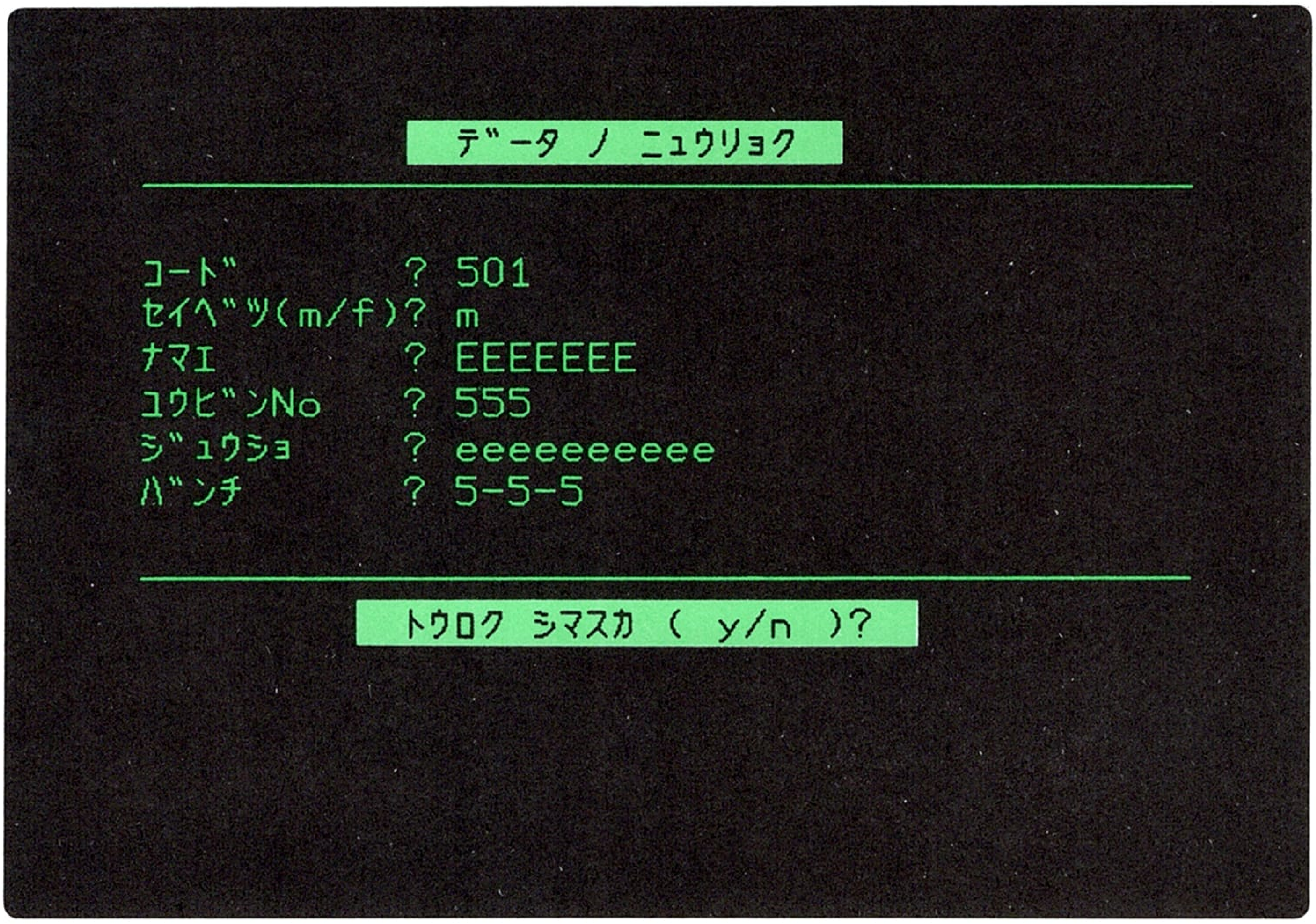
入力するデータですが、次の5人分とします。

コード	性別	氏　名	郵便番号	住　所	番　地
101	m	AAAAAAA	111	aaaaaaaaaa	1－1－1
201	m	BBBBBBB	222	bbbbbbbbbb	2－2－2
301	f	CCCCCCC	333	cccccccccc	3－3－3
401	f	DDDDDDD	444	dddddddddd	4－4－4
501	m	EEEEEEE	555	eeeeeeeeee	5－5－5

データといっても、氏名、住所がアルファベットでちょっと実感がわきませんが、このプログラムが確実に動作するかを確認するだけですから、入力しやすいほうがよいでしょう。

さあ、それでは f・5 キーを押してプログラムの実行です。

すると24ページのような初期画面になりますから、メッセージに従って上のデータを入力してください。次に示すのは5人目の人のデータを入力し終った場面で、登録するかどうかをうながしています。



ここで「Y」を入力すると、５人目の人のデータがディスクに書き込まれ、再びコード入力画面に戻ります。

私たちはこれで入力は終りですので「9999」を入力しリターンキーを押しましょう。

すると、画面に次のエラーメッセージが表示されてプログラム処理がストップしてしまいました。

```
File not found
Ok
■
```

なぜだかわかりますね。そうですね、まだ**メニュープログラム**

menu1.bas

を作っていないので、これが見つからないと文句を言っているのです。文句を言っててもかまいません。そっとこのままにしておきましょう。

なお、何かのつごうで**データ入力を最初からやりなおしたいときは、** STOP キーでプログラムの実行を止め、ダイレクトモードでファイルをCLOSEし、ランダムファイル、シーケンシャルファイルを**KILL（キル）コマンド**で削除してから再度RUNしてください。

KILLコマンド

次の画面は、ファイル削除の手続きを示しています。

```
close
Ok
kill "2:indx01.dat"
Ok
kill "2:mail01.dat"
Ok
■
```

1・4　書き込みデータの確認

前節でデータの入力が終りましたが、正確に書き込まれているかどうか、ちょっと不安ですね。

この節では、一時しのぎの方法で確認してみることにします。

確認の手順

シーケンシャルファイルについては、まずダイレクトモードでファイルをOPENし、FOR〜NEXTループを使って5個のデータを読み出し、それをPRINT文で画面に表示させます。

ランダムファイルについても手順は同じで、ファイルをOPENして必要なセクタをGETすればよいのですが、フィールド設定などがあり、ダイレクトモードでキーインするにはちょっとシンドイので、私たちが作ったプログラムの一部を流用することにします。

シーケンシャルファイルのデータ確認

まず、ダイレクトモードで次のようにキーインし、**シーケンシャルファイルをINPUT（読み出し）モードでOPEN**します。現在入力してあるプログラムは消さずにそのままにしておいてください。

```
open "2:indx01.dat" for input as #1 ⏎
                        └──データ読み出しモード
```

アクセス表示用の赤ランプがついて、インデックス用のシーケンシャルファイルがオープンされました。

「Ok」が表示されたら、ダイレクトモードで次のようにキーインします。

```
for i=1 to 5:input #1,cd:? cd;"   ";:next
             データ読み出し   データを表示
改行せずに表示させるために入れる ──────┘
```

どうですか？　リターンキーを押すやいなやデータが表示されましたね。そしてそれは、まぎれもなく私たちが入力したコードです。

これで、コードは無事読み込めていることがわかりましたので、ファイルをクローズしましょう。次のようにキーインします。

```
close #1 ⏎
```

それではダイレクトモードでキーインした結果を画面表示で示しておきます。同じになりましたか？

```
open "2:indx01.dat" for input as #1
Ok
for i=1 to 5:input #1,cd:? cd;"  ";:next
 101     201     301     401     501
Ok
█
```

ランダムファイルのデータ確認

次はランダムファイルの場合ですが、今度は、プログラムモードとダイレクトモードの混合で行ないます。

最初に、現在入力してあるプログラムの不要部分を削除します（**プログラムを消してしまった人は、「dmail1.bas」をロードしてください**）。次のようにキーインしてください。

不要行の削除

```
delete 100-840 ⏎
```

キーインできましたら、LISTコマンドでプログラムリストを画面に表示させてみてください。

行番号1000行以降のファイルオープンとフィールド設定のステートメントだけが残りましたね。

ランダムファイルではフィールド設定がありますから、ダイレクトモードでキーインするのはめんどうなので、ステートメントがせっかくそろっているのですから、これを流用します。

ただし、この部分はサブルーチンになっていますから、1070、2070行のRETURN文は削除しておきます。その方法は、**削除したい行番号を入力してリターン**でしたね。

```
1070 ⏎
2070 ⏎
```

さあ、もう一度プログラムリストを表示させてみてください。

```
list
1000 '―――――――――――――――――――――――
1010 *FOPEN1
1020 '―――――――――――――――――――――――
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
         6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1080 '
2000 '―――――――――――――――――――――――
2010 *FOPEN2
2020 '―――――――――――――――――――――――
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
Ok
■
```

リストをごらんください。1000行から、および2000行からのプログラムでうまくファイルをOPENできそうですね。

f・5 キーを押して実行してみましょう。

おやッ、**エラーメッセージ**がでてしまいました。

```
run
Subscript out of range in 2050
Ok
■
```

〈注〉CLEAR文で配列変数などを消去してから配列宣言をしないと、配列二重定義のエラーになります。

2050行で**配列の添字（そえじ）不足**ということですね。

上のリストをごらんください。2040行のFOR文でＩは256回ループすることになっています。したがって、配列も256の大きさで宣言しておかなければなりません。ダイレクトモードで、まず**ファイルをクローズし、次にCLEAR（注）してから配列を宣言します。**

画面は次のようになります。

```
clear
Ok
dim sex$(256)
Ok
█
```

さあ、これでだいじょうぶだと思いますが、**今度も前と同じように** **f・5** **キーでRUNさせると、同じ配列の添字不足のエラーになってしまいますよ。**

なぜだかわかりますか？　そうですね、RUNコマンドを実行するとすべて初期状態からスタートすることになるのでした。ですから、せっかく宣言した配列も無効になってしまうのです。

しかたがないですから、GOTO文を使います。次のステートメントをキーインしてください。

```
go to 1000 ⏎
```

アクセス表示用の赤ランプがついてランダムファイルがオープンされました。今度はエラーになりませんでした。

それでは最初に、第1セクタに書き込まれているはずの性別用のフラグを調べてみましょう。次のようにキーインし、**第1セクタをGET文でバッファ上に取り出します。**

〈注〉第1セクタをGETするとは、指定したファイルの第1セクタの内容をバッファ上に取り出すことです。このときその内容はフィールド定義された変数に割り当てられます。

```
get #2,1 ⏎
```

「Ok」が表示されましたら、続いて次のようにキーインしてください。

```
for i=1 to 5:? sex$(i);"  ";:next i ⏎
```

5人分のデータを入力したので、FOR～NEXTで5回ループさせ、バッファ上の配列変数の第1番目の要素から第5番目までの要素を見ようというわけです。

リターンキーを押すと、画面は次のようになりました。

```
go to 1000
Ok
get #2,1
Ok
for i=1 to 5:? sex$(i);"  ";:next i
m  m  f  f  m
Ok
■
```

どうですか？ うまく書き込まれていましたね。

私たちが入力した順に**「m　m　f　f　m」**と表示されました。

次は第2セクタ以降の様子を調べますが、全部をチェックしないで抜き取り検査式に1つだけ確かめてみましょう。たとえば36ページのデータで、コード番号401の人について調べてみましょう。

この人のデータは**4番目に入力しましたから、第3セクタの第1レコードになっている**はずですね。ということは、**第3セクタをGETして、フィールド設定された変数の添字1の内容を見ればよいことになります。**第2セクタ以降は1番バッファでファイルをOPENしていますから、次のようにキーインします。**バッファ番号を間違えないようにしてください。〈注〉**

〈注〉こういったテストをしたり、あるいはプログラムの動作確認をしたりしていると、とんでもない数字とかキャラクタが現れることがありますが、そんなときは取り出すセクタが違ったりバッファ番号が違ったりしている場合が多いようです。

```
get #1,3 ⏎
```

「Ok」が表示されたら、続いて次のようにキーインしてください。

```
? nm$(1):? pn$(1):? ad$(1):? ba$(1)
```

リターンキーを押すと画面は次のようになりました。

```
get #1,3
Ok
? nm$(1):? pn$(1):? ad$(1):? ba$(1)
DDDDDDD
444
dddddddddd
4-4-4
Ok
■
```

4人目の人の項目

ランダムファイル第3セクタの第1レコードの内容を表示

ーランダムファイルへの書き込みも正常に行われていたようです。

前に、バッファ２番においてダミーＤ＄も活躍の権利をもっていると述べました。
次のようにキーインしてみてください。

```
get #2,1 ⏎
```

２番バッファへ第１セクタの内容を取り出しました。そこでＤ＄には何が割り当てられているか次のようにして確認してみましょう。

```
? d$ ⏎
```

どうでしょうか？　画面に第１セクタの内容、

mmffm

が表示されたと思います。
現在５名のデータしか書き込んでありませんでしたからこれでよいのです。100人のデータを書き込んであれば100個の性別用フラグが表示されるはずです。
ところで、Ｄ＄は1000行からのルーチンでもダミーとして使われていますが、後から使われている2000行からの場合が優先されますので、**Ｄ＄に割り当てられているフィールドは２番バッファにおいて255バイト**ということになります。１番バッファにおいてはＤ＄は有効ではありません。
さて、Ｄ＄が本当に255バイトの大きさだけあるのかどうか調べてみましょう。
それには**LEN（レングス）関数**を使います。これは文字列の長さを調べる関数です。 LEN関数
次のように入力してください。

```
? len(d$) ⏎
```

255という数字が表示されましたね。これでＤ＄の良さが255バイトであることがわかりました。最後にファイルを**CLOSE**します。

```
close ⏎
```

本節を終るにあたって一連の手続きを示しておきます。ただしシーケンシャルファイルについては割愛してあります。

```
run
Subscript out of range in 2050   ← 配列の添字不足
Ok
close
Ok
clear   ← 変数をクリア
Ok
dim sex$(256)   ← 配列宣言
Ok
go to 1000
Ok
get #2,1   ← 2番バッファへ第1セクタの内容を取り出す
Ok
for i=1 to 5:? sex$(i);"   ";:next i
m   m   f   f   m   ← 性別用フラグ
Ok
get #1,3   ← 1番バッファへ第3セクタの内容を取り出す
Ok
? nm$(1):? pn$(1):? ad$(1):? ba$(1)
DDDDDDD     ┐
444         │
ddddddddd   ├ 4人目の人のデータ
4-4-4       ┘
Ok
get #2,1   ← 2番バッファへ第1セクタの内容を取り出す
Ok
? d$
mmffm   ← ダミーD$の内容
Ok
? len(d$)   ← ダミーD$の長さは？
 255   ← 255バイト
Ok
close   ← すべてのファイルをクローズ
Ok
█
```

1・5　データの追加

私たちが1・2節で作ったプログラムは、初回のデータ書き込みしかできません。これではちょっと不便ですね。ディスクに記憶されているデータを修正したいとか、あるいはあらたに追加したいとか、いろいろ希望があるはずです。

そこでこの節では、33〜35ページのプログラムリストに必要なステートメントを追加して、初回および追加時兼用のプログラムにしてみましょう。

それではまず、

dmail1.bas

をロードしてください。

ところであなたはプリンタをお持ちですか？　もし、プリンタの用意があるのでしたら、次のようにキーインしてプログラムリストをプリンタに出力し、印刷されたリストを見ながら読み進んでください。

LLIST ⏎　　　LLIST

プリンタをお持ちでない方は、しかたないですから、33〜35ページのリストか、ディスプレイ上のリストを見ながら進んでください。

初回の書き込み？　それとも追加？

最初に、初めてのデータ入力か、それとも既存のファイルに追加するのか判定しなければなりませんね。

この方法は、前編をお読みになった方はおわかりと思いますが、ランダムファイルをOPENして、その**ファイル長をLOF（レングス・オブ・ファイル）関数で調べ、もしその値が0ならば**まだ一度も書き込みが行なわれていないわけですから、**インデックス用のシーケンシャルファイルをOUTPUTモードで、その他の場合にはAPPENDモードでOPENします。**　　　LOF関数　OUTPUTモード　APPENDモード

データを追加したいのにOUTPUTモードでOPENすると、ファイルポインタがシーケンシャルファイルの先頭にセットされ、以前に書き込んだデータが無効になってしまいます。

そして、**データ追加の場合には、既存のファイルをINPUTモードでOPENし、書き込んであるデータ数を調べます。**この数がわかれば、最初の追加データが何番目のものか決まりますから、これによって、　　　INPUTモード

ランダムファイルへの書き込み位置が換算されます。

この部分のステートメントは、ランダムファイルをOPENしてからですから、210行のGOSUB文でサブルーチンに飛び、帰ってきたところへ入れます。

プログラムリストを次に示します。

```
211 IF LOF(2)=0 THEN GOTO 220
212 GOSUB *YOMU : P=COUNT
213 OPEN "2:indx01.dat" FOR APPEND AS #3
214 GOTO 240
```

OUTPUTモード

行番号211で、もし**2番のランダムファイル長が0**であれば、行番号220の**OUTPUTモード**へ飛ばします。

追加の場合には212行へ処理が落ちてきますから、サブルーチンへジャンプさせ、すでに書き込みずみのデータ数をカウントします。

この数値は変数Pに代入され、行番号560でカウンタを1上げていますから、ランダムファイルには既存のデータの次から書き込まれることになります。

そして行番号213で、シーケンシャルファイルを**APPENDモード**でOPENします。

APPENDモード

それでは、行番号212の飛び先であるサブルーチンを作っておきましょう。プログラムは次のとおりです。

```
3000 '―――――――――――――――――――――――――――――――
3010 *YOMU
3020 '―――――――――――――――――――――――――――――――
3030 OPEN "2:indx01.dat" FOR INPUT AS #3
3040 WHILE NOT EOF(3)
3050   COUNT=COUNT+1 ←―――――――データ数をカウント
3060   INPUT #3,DUMMY ←――――――読み飛ばし
3070 WEND
3080 CLOSE #3
3090 RETURN
3100 '
```

WHILE〜WEND

3040行と3070行の**WHILE〜WEND文**はわかりますね。

WHILEのうしろの条件が満たされている間は、WENDとの間のステートメントを実行します。この場合、シーケンシャルファイルの末

尾のデータを読み出すまでループします。

この部分は、データ数を数えるだけですから、3050行でデータを読むときも変数はダミーでいいですね。

そして、RETURN文でもとに戻るときに、ファイルをクローズするのを忘れないでください。

これ以上データを書き込めない！

このプログラムでは、性別を見分けるフラグ用として第1セクタだけを使っていますので、書き込むことができるデータ数は256までです。

したがって、**行番号560のカウンタ用変数Pの内容が、256を超えてしまうとまずいわけです。**ですから、行番号を565として制限をつけておくことにします。

次のようにキーインしてください。

```
565 IF P>256 THEN GOSUB *ACCIDENT
```

万一このようなことが起こったら、プログラムのうしろのほうへ飛ばして、一時的に処理をしておきましょう。

このような事故を防ぐためには、できるだけ長期的な立場に立って考え、第2セクタ、第3セクタをもフラグ用に確保しておくとよいと思います。

次に565行からのジャンプ先のサブルーチンを示します。

```
4000 '―――――――――――――――――――――――――
4010 *ACCIDENT
4020 '―――――――――――――――――――――――――
4030 LOCATE 20,11 : PRINT SPC(40)
4040 LOCATE 30,11 : COLOR 6
4050 PRINT "  ﾄｳﾛｸ ﾃﾞｷﾏｾﾝ !  "
4060 BEEP : BEEP : BEEP
4070 FOR I=1 TO 5000 : NEXT I
4080 COLOR 0 : RETURN *OWARI
```

説明は必要ないと思いますが、

トウロク　デキマセン　！

というエラーメッセージを反転ブリンクさせて表示し、**RETURN文**で終りの処理へ飛ばしています。つまりこの場合、メニュープログラム

RETURN文
●RETURN文はこのように戻り先を、行番号あるいはラベルで指定できます。

へ戻ってしまいます。

プログラムの実行

〈注〉前節と同じファイル名でセーブすると前のプログラムは消えてしまいますが、この節でのプログラムが正常に走れば、前のものは使う機会を失いますのでかまわないでしょう。

さあ、これでだいたいプログラムの修正がすみましたので、これを実行して動作テストをしてみましょう。

その前に、**このプログラムを一応セーブしておいてください。**ファイル名は前と同じでかまいません。〈注〉

```
save "dmail1.bas"
```

セーブできましたら、前と同じように、次の2人分のデータを追加してみましょう。

```
601   f   FFFFFFF   666   ffffffffff   6-6-6
701   m   GGGGGGG   777   gggggggggg   7-7-7
```

これは、6人目と7人目のデータになりますね。入力方法はもうわかりますね。RUNさせてメッセージに従ってキーインしてください。

入力が終りましたら、39ページで学んだ方法でデータの確認をしてみます。まず、次のように不要な行をデリートします。

```
delete 100-840
Ok
delete 3000-4080
Ok
1070
2070
■
```

そうしたら、右ページのようにダイレクトモードでキーインし、**7つ目のデータが書き込まれる第4セクタをGETしてみます。**

〈注〉ランダムファイル上では8人目、9人目の書き込み位置にデータが書かれてしまったことになりますが、インデックスには登録されていませんから別に支障はないのですが、気分的にスッキリしませんね。

右ページの画面をごらんください。

7人目のデータまで入力したのですから、第4セクタは1番目のレコードである7人目のデータ以外は書き込まれてはいけないわけです。しかし画面では、5人目、6人目のデータが第2レコード、第3レコードとして表示されています。〈注〉

```
clear ←――――――――――――――――――――――変数をクリア
Ok
dim sex$(256) ←――――――――――――――――――配列宣言
Ok
go to 1000 ←――――――――――――――――――――1000行から実行
Ok
get #1,4 ←―――――――――――――――――――――第 4 セクタをGET
Ok
for i=1 to 3:? nm$(i);pn$(i):next
GGGGGGG                777 ←――――――――7 人目のデータ
EEEEEEE                555 ┐
FFFFFFF                666 ┘――――5 人目、6 人目のデータが書き込まれている
Ok
close ←―――――――――――――――――――――――ファイルをCLOSE
Ok
█
```

このようになってしまった原因を知るのに、プログラムリストの必要な箇所をもう一度示します。

まず**プログラムをもう一度ロード**してください。そして**LISTコマンド**で必要箇所を表示してみましょう。

```
load "dmail1.bas"⏎
```

```
list 570-660
570 SEC=(P-1) ¥ 3+2
580 REC=(P-1) MOD 3+1
590 IF REC>1 THEN GET #1,SEC : GET #2,1
600 WRITE #3,CD
610 LSET SEX$(P) =A1$
620 LSET NM$(REC)=A2$
630 LSET PN$(REC)=A3$
640 LSET AD$(REC)=A4$
650 LSET BA$(REC)=A5$
660 PUT #1,SEC : PUT #2,1
Ok
█
```

590行で、第1レコードの書き込みであったら、THEN文以下のGET文を通さないで600行へ処理を移し、610行以下でデータをバッファ上でLSETしています。

しかし、ここで注意してください。このとき、第1レコードのデータは確実にセットされますが、第2レコード、第3レコードのバッファ上の変数には何が入っているでしょうか？

7人目の1つ前、6人目のデータをセットするとき、**590行で第3セクタをGETしていますから、このときの状態が、7人目のデータ書き込みのときにもそのまま持ち込まれています。**

つまり、7人目のデータを書き込むときも、第2レコード、第3レコード用のバッファ上の変数に、第3セクタの第2レコード、第3レコードのデータが保存されていたのです。

変数のクリア

ですから、**第1レコードの書き込みの場合には、LSETする前に、第2レコード、第3レコード用の変数をバッファ上でクリアしておかなければならないのです。**

バッファクリア

これと同じことが第2レコードの書き込みの場合にもいえます。このときは**第3レコード用変数をバッファ上でクリアする**ことになります。

2つの場合を考慮して次のステートメントを追加します。

```
595 IF REC<3 THEN GOSUB *BUFCLEAR
```

続いてサブルーチンも入力してしまいましょう。

```
4090 '
5000 '————————————————————————————————
5010 *BUFCLEAR
5020 '————————————————————————————————
5030 FOR I=REC+1 TO 3
5040   LSET NM$(I)=""  ┐
5050   LSET PN$(I)=""  ├──バッファ上で変数をクリア
5060   LSET AD$(I)=""  │
5070   LSET BA$(I)=""  ┘
5080 NEXT I
5090 RETURN
```

この場合、ループ変数Iの初期値は、RECの内容が1あるいは2のいずれかになりますから、2あるいは3になります。つまり、第2または第3レコード用の変数の内容をバッファ上でクリアすることになります。

また、5040行から5070行までのダブルクォーテーションの間には何も文字列が入っていません。つまり、**ヌルストリング**です。

これでうまく動作すると思います。

次の3名のデータを追加してみたいと思いますが、その前にまず**プログラムをセーブ**してください。

801	m	HHHHHHH	888	hhhhhhhhhh	8-8-8
901	f	IIIIIII	999	iiiiiiiiii	9-9-9
001	f	JJJJJJJ	000	JJJJJJJJJJ	0-0-0

データ入力が終わりましたら、もう一度、39～42ページの方法で、今度は第5セクタをGETして調べてみてください。

```
delete 100-840
Ok
delete 3000-5090
Ok
1070
2070
clear
Ok
dim sex$(256)
Ok
go to 1000
Ok
get #1,5
Ok
for i=1 to 3:? nm$(i);pn$(i):next
JJJJJJJ                  000
                               ] 何も書き込まれていない

Ok
```

今度は第5セクタの第2、第3レコードには何も書き込まれていませんでしたね。

コード番号001の人の項目だけが書き込まれていました。

また、10人目の人のコードは001となっていますが、これは他と桁を合わせるためで、入力は1をキーインすればかまいません。

●8人目、9人目のデータは、すでにデータが書き込まれている第4セクタの第2、第3レコードの位置に重ねて書き込まれ、前のデータは消失します。

RENUMコマンド

それではRENUMコマンドで行番号を整理しておきましょう。もう一度プログラム

dmail1.bas

をロードして、ダイレクトモードで次のようにキーインしてください。

```
renum 100
Ok
renum 1000,910
Ok
renum 2000,1090
Ok
renum 3000,2090
Ok
renum 4000,3110
Ok
renum 5000,4100
Ok
■
```

リナンバリングが終ったら、**もう一度プログラムをセーブしなおしておく**のを忘れないでください。

次に、プログラムの総リストを示します。

```
100 ' file name ････> dmail1.bas
110 ' 1985.2.10
120 ' ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｶｷｺﾐ
130 ' save "dmail1.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
220 IF LOF(2)=0 THEN GOTO 260
230 GOSUB *YOMU : P=COUNT
240 OPEN "2:indx01.dat" FOR APPEND AS #3
250 GOTO 280
```

```
260 OPEN "2:indx01.dat" FOR OUTPUT AS #3
270 '
280 COLOR 4
290 LOCATE 31,0 : PRINT "  ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ  "
300 COLOR 0
310 LOCATE 20,1 : PRINT STRING$(40,149)
320 LOCATE 20,10: PRINT STRING$(40,149)
330 '
340 '――――――――――――――――――――――――――――――――――――
350 *LOOP
360 '――――――――――――――――――――――――――――――――――――
370 COLOR 6
380 LOCATE 40,3 : PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  "
390 COLOR 0
400 LOCATE 20,3 : INPUT "ｺｰﾄﾞ        ";CD
410 IF CD=9999 THEN GOTO *OWARI
420 LOCATE 40,3 : PRINT SPC(20)
430 PRINT TAB(20) : INPUT "ｾｲﾍﾞﾂ(m/f)";A1$
440 PRINT TAB(20) : INPUT "ﾅﾏｴ       ";A2$
450 PRINT TAB(20) : INPUT "ﾕｳﾋﾞﾝNo   ";A3$
460 PRINT TAB(20) : INPUT "ｼﾞｭｳｼｮ    ";A4$
470 PRINT TAB(20) : INPUT "ﾊﾞﾝﾁ      ";A5$
480 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
490 COLOR 4 : LOCATE 27,11
500 PRINT "   ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?    "
510 COLOR 0
520 I$=INKEY$ : IF I$="" THEN GOTO 520
530 IF I$="N" OR I$="n" THEN GOTO *KESU
540 IF I$="Y" OR I$="y" THEN GOTO *KAKU
550 BEEP : BEEP : GOTO 520
560 '
570 '――――――――――――――――――――――――――――――――――――
580 *KAKU
590 '――――――――――――――――――――――――――――――――――――
600 P=P+1
610 IF P>256 THEN GOSUB *ACCIDENT
620 SEC=(P-1) ¥ 3+2
630 REC=(P-1) MOD 3+1
640 IF REC>1 THEN GET #1,SEC : GET #2,1
650 IF REC<3 THEN GOSUB *BUFCLEAR
660 WRITE #3,CD
670 LSET SEX$(P) =A1$
```

```
680 LSET NM$(REC)=A2$
690 LSET PN$(REC)=A3$
700 LSET AD$(REC)=A4$
710 LSET BA$(REC)=A5$
720 PUT #1,SEC : PUT #2,1
730 '
740 '----------------------------------------------------------
750 *KESU
760 '----------------------------------------------------------
770 LOCATE 20,3
780 FOR I=1 TO 6
790   PRINT TAB(20);SPC(40)
800 NEXT I
810 LOCATE 20,11 : PRINT SPC(40)
820 GOTO *LOOP
830 '
840 '----------------------------------------------------------
850 *OWARI
860 '----------------------------------------------------------
870 CLS
880 CLOSE
890 RUN "menu1.bas"
900 '
1000 '---------------------------------------------------------
1010 *FOPEN1
1020 '---------------------------------------------------------
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
         6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '---------------------------------------------------------
2010 *FOPEN2
2020 '---------------------------------------------------------
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
2080 '
```

```
3000 '-----------------------------------------------
3010 *YOMU
3020 '-----------------------------------------------
3030 OPEN "2:indx01.dat" FOR INPUT AS #3
3040 WHILE NOT EOF(3)
3050   COUNT=COUNT+1
3060   INPUT #3,DUMMY
3070 WEND
3080 CLOSE #3
3090 RETURN
3100 '
4000 '-----------------------------------------------
4010 *ACCIDENT
4020 '-----------------------------------------------
4030 LOCATE 20,11 : PRINT SPC(40)
4040 LOCATE 30,11 : COLOR 6
4050 PRINT "   ﾄｳﾛｸ ﾃﾞｷﾏｾﾝ !   "
4060 BEEP : BEEP : BEEP
4070 FOR I=1 TO 5000 : NEXT I
4080 COLOR 0 : RETURN *OWARI
4090 '
5000 '-----------------------------------------------
5010 *BUFCLEAR
5020 '-----------------------------------------------
5030 FOR I=REC+1 TO 3
5040   LSET NM$(I)=""
5050   LSET PN$(I)=""
5060   LSET AD$(I)=""
5070   LSET BA$(I)=""
5080 NEXT I
5090 RETURN
```

1・6　データの修正

データインプットのプログラムが完成しましたので、次は、すでにディスクに書き込んであるデータを修正したい場合を考えてみましょう。

前編でも取り上げましたので、どうすればよいかすでに目星がついている方もいらっしゃるかもしれません。

しかし、今回は、ランダムファイルの異なるセクタに同時に書き込むという点で前回とは多少違いますし、また同じようなことを何回もくりかえすことが、プログラミングマスターへの着実な道だと思いますので、ぜひおつきあいしてください。

プログラムの流れ

データ修正の手順

データ修正の基本的な手順は、まず第１にインデックス用のシーケンシャルファイルからデータを読み込み、それを配列にしまいます。次に、修正したい人のコードを入力し、そのコードによってランダムファイルからデータを取り出してきて、画面で確認しながら新しいデータを入力していきます。

そして、修正データをキーインし終ったら、ランダムファイルの該当セクタに書き込みます。

以上が、データ修正のおおまかな流れです。

以上を頭に描いておいていただいた上で、画面がどのように流れるか簡単に説明しておきましょう。

❶　プログラムを実行すると図１のようなコード入力画面になります。

❷　コードを入力すると配列の中を検索し、該当コードがない場合は図２のようにエラーメッセージが表示されます。

❸　もし該当コードがあれば図３のように修正データの氏名を画面に表示し、修正するかどうかの確認をうながします。

❹　確認メッセージに対して「Y」を入力すれば図４のようにデータの修正を行うことになりますし、もし「N」を入力するともう一度コード入力からやりなおします。

　なおデータの修正は、すでにディスクに書き込みずみの項目が１つずつ表示されますから、もし修正がなければリターンキーを押してください。修正したい項目のみキーボードより正しいデータを入力します。

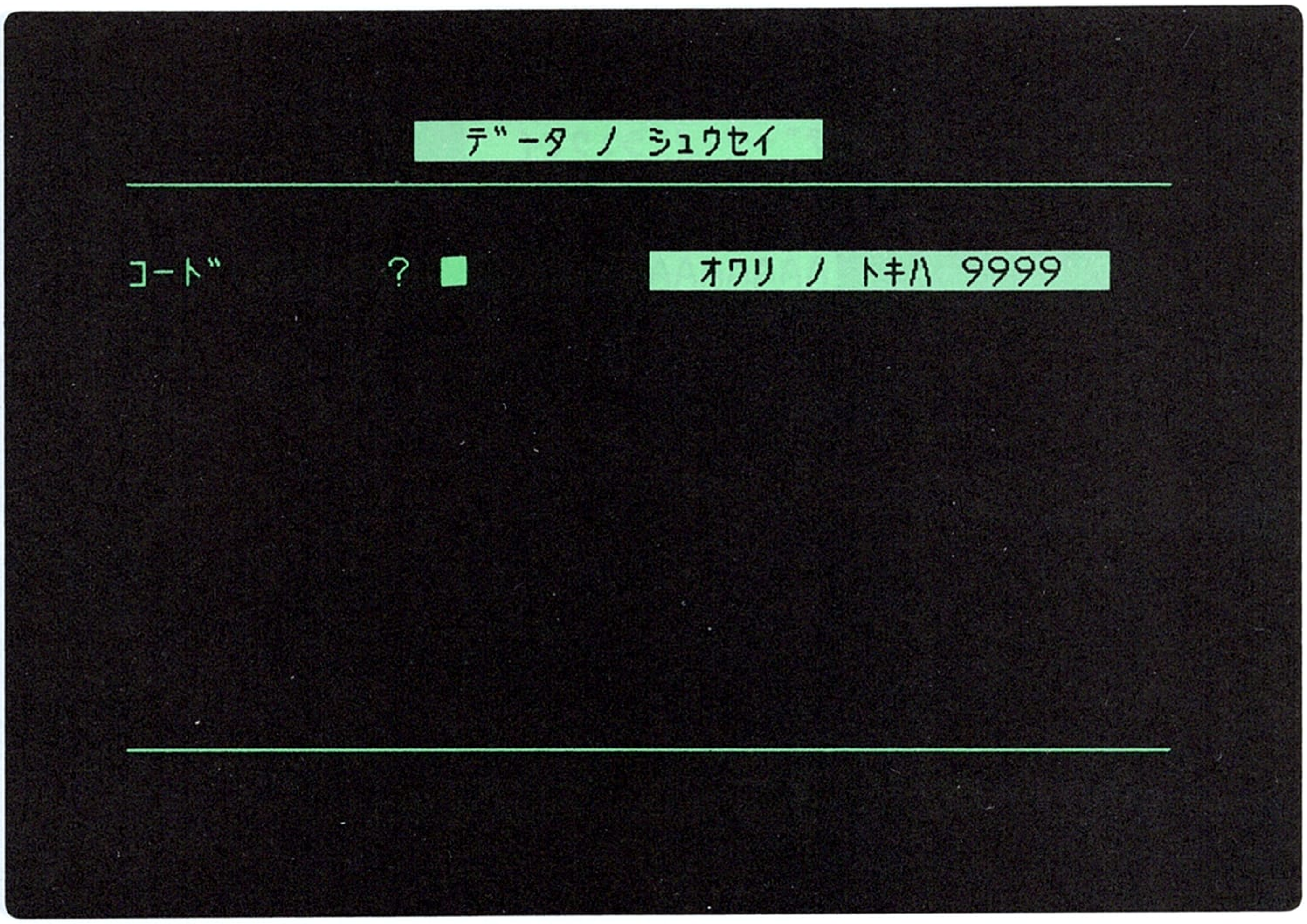


図1●修正データのコード入力

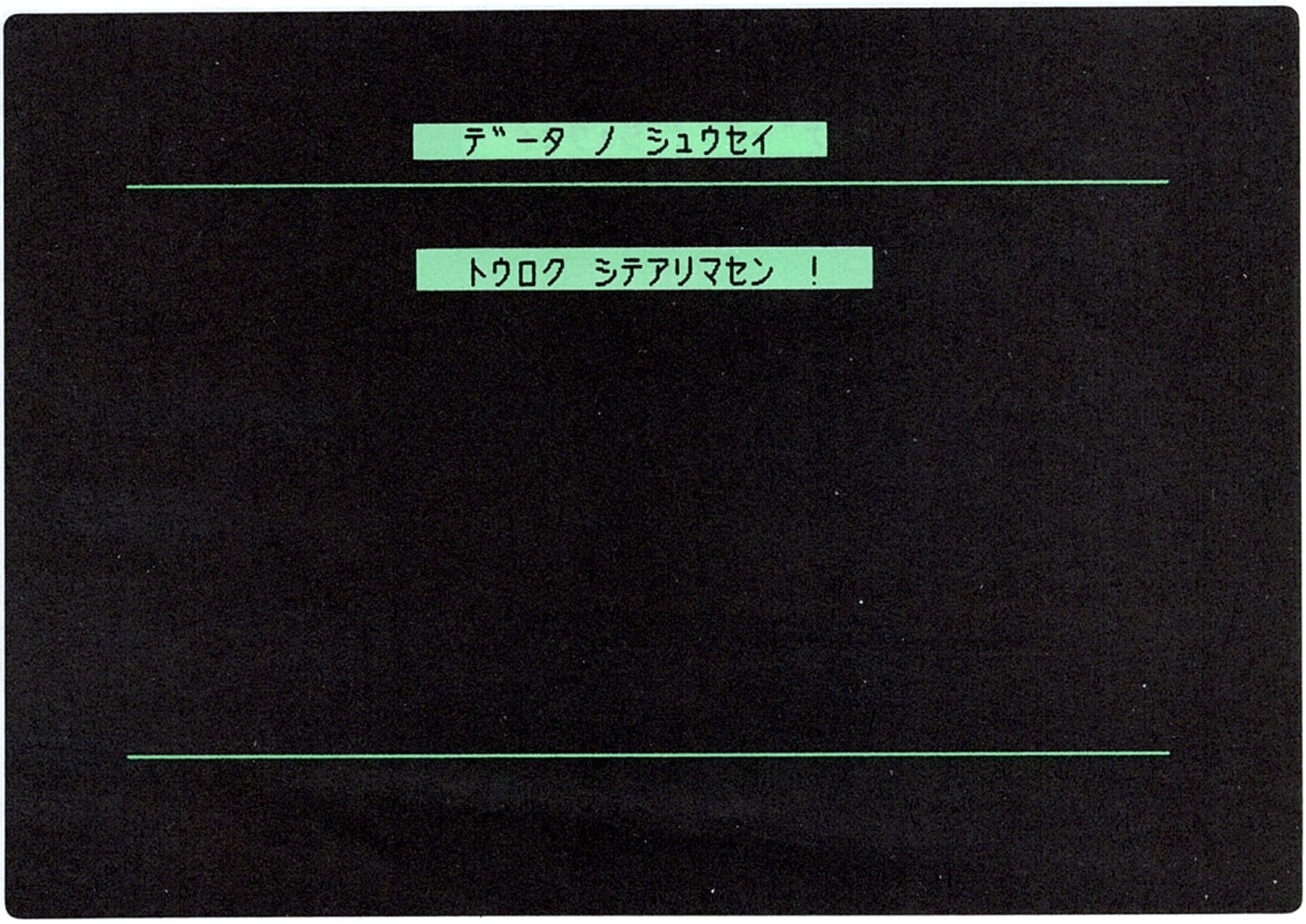


図2●該当コードがないときのエラーメッセージ

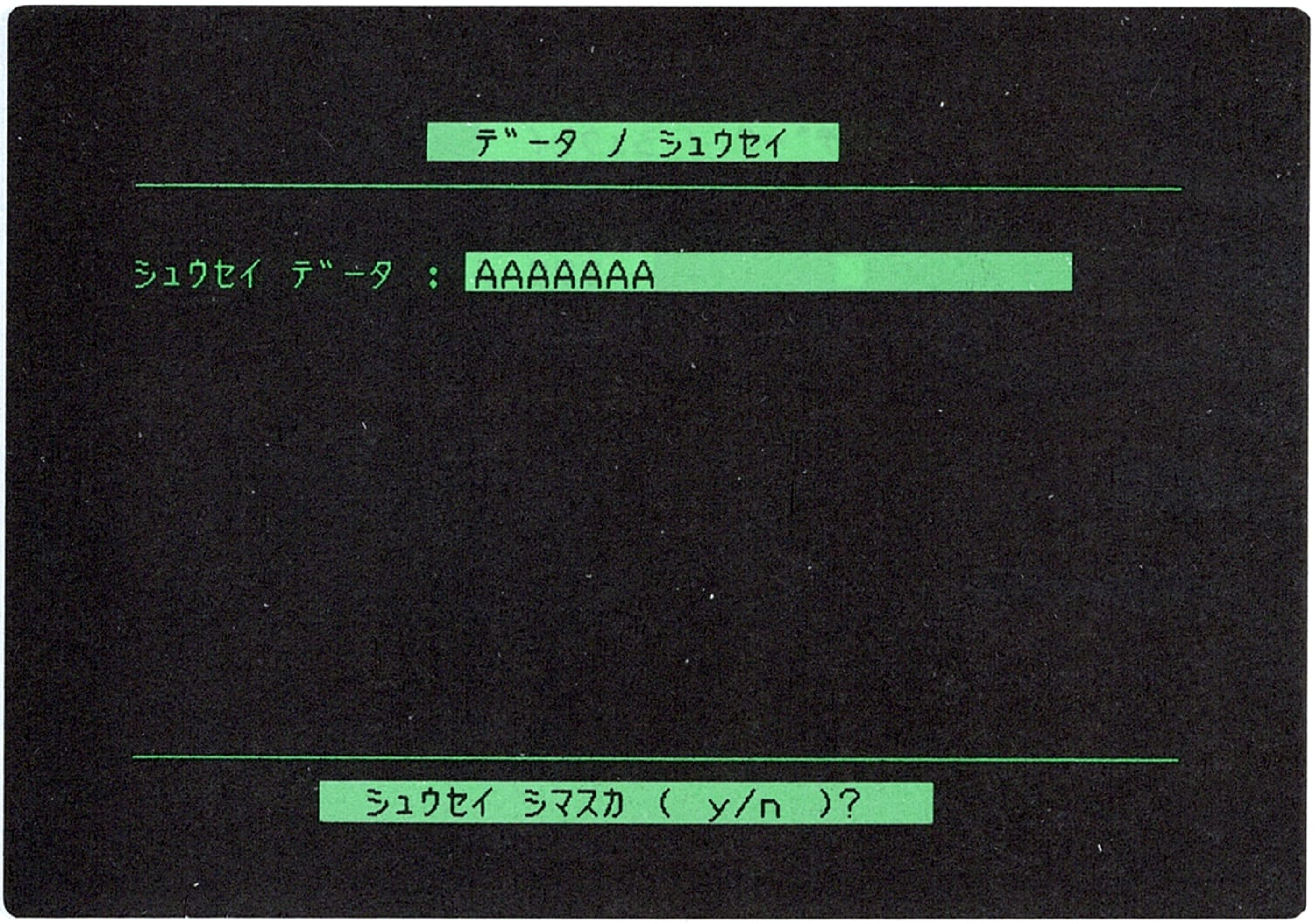


図 3 ●修正するかどうかの確認画面

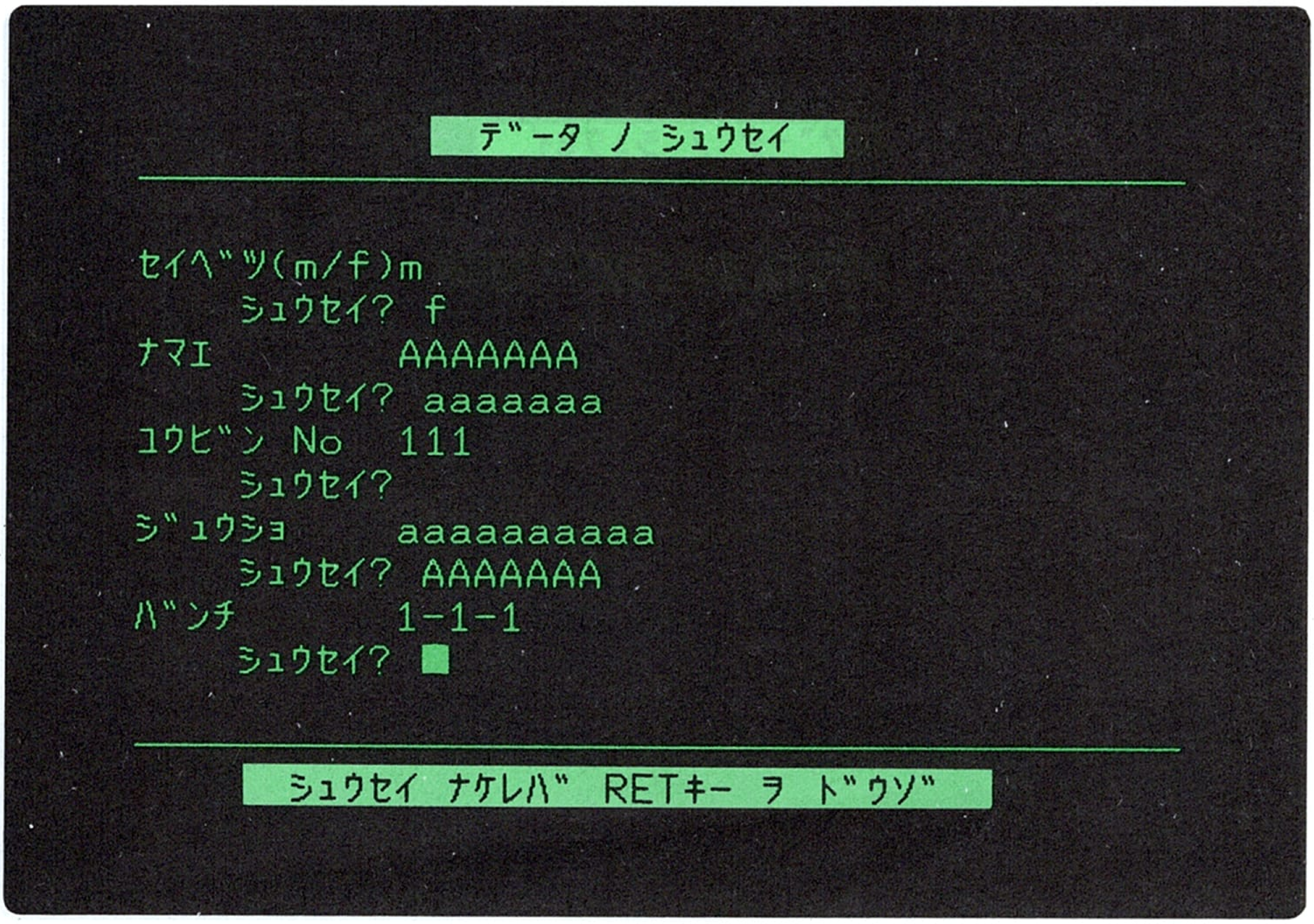


図 4 ●データの修正中

初期設定からファイルオープンまで

まずプログラムの入口部分です。次を入力してください。

```
100 '  file name ····> dmail2.bas
110 '  1985.2.10
120 '  ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ
130 '  save "dmail2.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
```

これはいつもと同じですからわかりますね。このプログラムのファイル名を間違わないようにしてください。

200行は第2セクタ以降のための、210行は第1セクタのための**ファイルオープン**と**フィールド定義**です。

```
1000 '————————————————————————————————
1010 *FOPEN1
1020 '————————————————————————————————
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '————————————————————————————————
2010 *FOPEN2
2020 '————————————————————————————————
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
2080 '
```

サブルーチンの意味も問題ないですね。前節とまったく同じです。

配列にコードを読み込む

次のプログラムを入力してください。

```
220 GOSUB *YOMU : DIM CD(COUNT) : COUNT=0
230 NIKAIME=-1 : GOSUB *YOMU
240 '
```

```
3000 '────────────────────
3010 *YOMU
3020 '────────────────────
3030 OPEN "2:indx01.dat" FOR INPUT AS #3
3040 WHILE NOT EOF(3)
3050   COUNT=COUNT+1
3060   IF NIKAIME THEN GOTO 3080 ←── 2回目の場合は3080行へ
3070   INPUT #3,DUMMY : GOTO 3090 ←── 1回目のときはカラ読み
3080   INPUT #3,CD(COUNT)
3090 WEND
3100 CLOSE #3
3110 RETURN
3120 '
```

220行でシーケンシャルファイル読み出しのサブルーチンへジャンプし、**データのカラ読み**をして**データ数をカウント**します。

1回目のサブルーチンへのジャンプのときは、変数NIKAIMEの内容は0ですから、3060行の

IF NIKAIME THEN~

にはひっかかりませんから、3070行のカラ読みを実行します。これがファイルが終るまで(**End Of File**となるまで)繰り返されます。

そしてRETURN文で飛び先へ帰ったときに、このデータ数(変数COUNTに格納されている)を添字(そえじ)としてコード格納用配列CDを宣言します。次にCOUNTをゼロクリアしてもう一度サブルーチンへジャンプし、コードを配列内にしまいますが、このとき、

NIKAIME=−1

とし、変数NIKAIME(「2回目」のジャンプだからこのようにしまし

た）を「**真**」としておきます。

こうすることにより、IF文の条件式NIKAIMEが「真」となりますからTHEN以下が実行され3080行へジャンプし（3060行）、配列CDにコードを読み込むことになります。

これでシーケンシャルファイルに書き込まれているコードはすべて配列に格納されました。

なおカウンタCOUNTのリセットを忘れると、プログラム完成後RUNさせた場合に、2回目のジャンプで、

```
Subscript out of range in 3080
```

のエラーになってしまいますから注意してください。

配列の添字不足ということですね。つまり、データ数を添字にして配列宣言をしたのはいいが、COUNTの内容がクリアされていないために、3050行でデータ数を基点にカウントされ3080行で宣言した添字を超えてしまうためです。

〈注〉「真」および「偽」については前編に詳しく解説してありますので、そちらを参照ください。

データの修正ルーチン

これで修正の準備ができましたので、次はこのプログラムの中心、修正ルーチンです。

まず、**入力画面の作成**です。次をキーインしてください。

```
250 COLOR 4
260 LOCATE 31,0 : PRINT "  デ゛ータ ノ シュウセイ   "
270 COLOR 0
280 LOCATE 20,1 : PRINT STRING$(40,149)
290 LOCATE 20,14: PRINT STRING$(40,149)
300 '
```

処理内容の表示と、**上下の罫線の出力**です。

次は修正したいデータのコード入力のメッセージ表示です。

```
310 '―――――――――――――――――――――――――――
320 *LOOP
330 '―――――――――――――――――――――――――――
340 COLOR 6
350 LOCATE 40,3 : PRINT " オワリ ノ トキハ 9999  "
360 COLOR 0
370 LOCATE 20,3 : INPUT "コート゛       ";CD
380 IF CD=9999 THEN GOTO *OWARI
```

修正作業を終りたいときはコードに「9999」を入力するようにしてあります。

そして、コードが入力されると、先ほど登録コードを格納した配列内をチェックするためにサブルーチンへジャンプさせます。

```
390 GOSUB *SAGASU
```

ある配列にしまわれているコード番号の中から、キーインしたデータを探し出すにはどうしたらよいかわかりますか？

SEARCH関数

これには大変便利な**SEARCH（サーチ）関数**がありますので、これを使ってみましょう。

まず、次のようにダイレクトモードでキーインしてください。

```
for i=1 to 5:c%(i)=i:next i   ← 整数配列C%に1から5までの数値を格納
Ok
a=5                           ← aに5を代入
Ok
? search(c%,3)                ← 3という数値が配列内にあるか探す
 3                            ← 3番目にありました
Ok
? search(c%,a)                ← aの内容5はあるか？
 5                            ← 5番目にありました
Ok
? search(c%,8)                ← 8はどうか
-1                            ← 見つからなかった
Ok
■
```

●SEARCH関数は、1次元の整数型配列にのみ有効です。カッコの中の数式の並びは、上の例では配列名、サーチする整数となっていますが、さらにこのうしろに開始ナンバー、ステップ値などを指定することもできます。

この画面をごらんいただければわかると思いますが、**C%(1)〜C%(5)までに1〜5の整数を代入し、SEARCH関数で3という整数がC%という配列内にあるかどうかサーチしています。**

サーチののち、画面に表示された「3」という値は、配列の要素番号を示していますから、C%(3)の中にしまわれていることが確認できるわけです。反対にもし該当データがない場合には、「−1」が返されます。

さあ、それではSEARCH関数を使ってサブルーチンを完成させましょう。次のようにキーインしてください。

```
4000 '
4010 *SAGASU
4020 '
4030 P=SEARCH(CD,CD)  ←配列内を探す
4040 IF P=-1 THEN GOSUB *MITOOROKU  ←該当コードなし
4050 SEC=(P-1) ¥ 3+2   ┐
4060 REC=(P-1) MOD 3+1 ┘←データのあるセクタ、レコード番号を算出
4070 GET #1,SEC : GET #2,1  ←セクタの内容をバッファへ
4080 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
4090 LOCATE 20,3 : PRINT SPC(40)
4100 LOCATE 20,3 : PRINT "ｼｭｳｾｲ ﾃﾞｰﾀ : ";  ┐
4110 COLOR 4 : PRINT NM$(REC) : COLOR 0     ├確認
4120 LOCATE 28,15 : COLOR 6                 ┘
4130 PRINT " ｼｭｳｾｲ ｼﾏｽｶ ( y/n )? "
4140 COLOR 0
4150 I$=INKEY$ : IF I$="" THEN GOTO 4150
4160 IF I$="Y" OR I$="y" THEN GOTO 4190
4170 IF I$="N" OR I$="n" THEN GOTO 4200
4180 BEEP : BEEP : GOTO 4150
4190 SHUUSEI=-1  ←修正する場合フラグを真に
4200 LOCATE 20,3 : PRINT SPC(40)
4210 LOCATE 20,15 : PRINT SPC(40)
4220 RETURN
4230 '
```

4030行のSEARCH関数のカッコ内の配列名および数式が、ともに同じCDとなっていますが、これらはまったく別のものですから注意してください。

また配列名に%がついていませんが、プログラムの先頭で

```
DEFINT A-Z
```

を行っていますので、つける必要はありません。

配列CDの中を探し、もしコードCDが見つからなければ**「未登録」**のサブルーチンへジャンプさせます。このサブルーチンはあとで示します。

該当コードがあった場合には、そのレコードが書き込まれているセクタ番号、およびそのセクタ内でのレコード番号を算出し、そのセクタをGETしてバッファ上に内容を取り出しておきます。これは**4050行〜4070行**で行っています。

次に、このコード番号に対応した氏名を画面に表示し、修正するか

どうかの確認をうながします。

もし「Y」と回答した場合は、4190行へジャンプさせて、

SHUUSEI=-1

としています。これはサブルーチンから帰ってから、**修正することに決まったのかどうかの判断用**に使います。

「Y」の場合でも「N」の場合でも、いずれにしても現在画面に表示されているメッセージを消去して（4200行、4210行）サブルーチンから戻ります。

では次に**該当コードがない場合のサブルーチン**をキーインしてしまいましょう。

```
5000 '─────────────────────────
5010 *MITOOROKU
5020 '─────────────────────────
5030 BEEP : BEEP : BEEP
5040 LOCATE 20,3 : PRINT SPC(40)
5050 LOCATE 28,3 : COLOR 6
5060 PRINT "  ﾄｳﾛｸ ｼﾃｱﾘﾏｾﾝ ! " : COLOR 0
5070 FOR I=1 TO 5000 : NEXT I
5080 LOCATE 20,3 : PRINT SPC(40)
5090 RETURN 4220
```

この場合ビープ音を鳴らして、

トウロク　シテ　アリマセン　！

と、**エラーメッセージ**を表示します。

RETURN文

戻り先は**RETURN文によって行番号を指定**しています。

さて、メインルーチンへ戻ったら、

該当コードなしの場合
該当データはあったが修正しない場合
修正する場合

の3つの場合分けをしなければなりません。

まず次のプログラムを入力してください。

```
400 IF P=-1 THEN GOTO *LOOP
410 IF NOT SHUUSEI THEN GOTO *LOOP
```

400行は

SEARCH関数で探しても該当コードがなかった場合

410行は

SHUUSEIの内容が「真でない」場合

つまり、

データは見つかったが「修正したくない」という場合

で、いずれにしても初めからやりなおしです。

そしてそれ以外の場合が、データ修正となります。次のプログラムを入力しましょう。

```
420 LOCATE 25,15 : COLOR 4
430 PRINT " ｼｭｳｾｲ ﾅｹﾚﾊﾞ RETｷｰ ｦ ﾄﾞｳｿﾞ"
440 COLOR 0
450 LOCATE 20,3
460 PRINT TAB(20) : PRINT "ｾｲﾍﾞﾂ(m/f)";SEX$(P)
470 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A1$
480 IF A1$="" THEN A1$=SEX$(P)
490 PRINT TAB(20) : PRINT "ﾅﾏｴ       ";NM$(REC)
500 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A2$
510 IF A2$="" THEN A2$=NM$(REC)
520 PRINT TAB(20) : PRINT "ﾕｳﾋﾞﾝ No  ";PN$(REC)
530 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A3$
540 IF A3$="" THEN A3$=PN$(REC)
550 PRINT TAB(20) : PRINT "ｼﾞｭｳｼｮ    ";AD$(REC)
560 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A4$
570 IF A4$="" THEN A4$=AD$(REC)
580 PRINT TAB(20) : PRINT "ﾊﾞﾝﾁ      ";BA$(REC)
590 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A5$
600 IF A5$="" THEN A5$=BA$(REC)
610 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
620 COLOR 4 : LOCATE 25,15
630 PRINT "      ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?     "
640 COLOR 0
650 I$=INKEY$ : IF I$="" THEN GOTO 650
660 IF I$="N" OR I$="n" THEN GOTO *KESU
670 IF I$="Y" OR I$="y" THEN GOTO *KAKU
680 BEEP : BEEP : GOTO 650
690 '
```

修正方法

まず、すでにディスクに書き込まれているデータを

セイベツ(m/f) m

のように表示し、続いてその下に

シュウセイ?

のプロンプト文を表示し、修正があるときはキーボードより入力し、**修正がなければリターンキー**のみ押すようにします。

リターンキーが押されると、ディスクに書き込まれていたデータがA１＄～A５＄に代入されます。

〈注〉本書では確認などの場合ＹまたはＮで答えさせますが、大文字でも小文字でも機能するようにしてあります。

５項目についての処理が終ると、

トウロク シマスカ (y/n)?

の確認メッセージが一番下のキーオペレーション行に表示されますから、ＹまたはＮで答えます。

「N」であれば画面を消去して初めからやりなおし、「Y」のときにはディスクへ新データを書き込みます。

データ書き込みルーチン

次を入力してください。

```
700 '―――――――――――――――――――――――
710 *KAKU
720 '―――――――――――――――――――――――
730 LSET SEX$(P) =A1$
740 LSET NM$(REC)=A2$
750 LSET PN$(REC)=A3$
760 LSET AD$(REC)=A4$
770 LSET BA$(REC)=A5$
780 PUT #1,SEC : PUT #2,1
790 '
```

バッファ上でLSET文により、フィールド定義された配列変数にセットし、PUT文でディスクに書き込んでいます。

これはもう皆さんおなじみの決まり文句です。

このルーチンを抜け出たら、データ表示行を消去して再びコード入力に戻ります。

次のプログラムを入力してください。

```
800 '―――――――――――――――――――――――――――――
810 *KESU
820 '―――――――――――――――――――――――――――――
830 LOCATE 20,3
840 FOR I=1 TO 10
850   PRINT TAB(20);SPC(40)
860 NEXT I
870 LOCATE 20,15 : PRINT SPC(40)
880 SHUUSEI=0
890 GOTO *LOOP
900 '
```

ループに戻る前に**変数SHUUSEIの内容をゼロクリアしておくのを**忘れないようにしてください。

終りの処理

最後に、コードに**「9999」**が入力された場合の処理です。

```
910 '―――――――――――――――――――――――――――――
920 *OWARI
930 '―――――――――――――――――――――――――――――
940 CLS
950 CLOSE
960 RUN "menu1.bas"
970 '
```

すべてのファイルをクローズし、メニュープログラムを実行させます。

さあ、プログラムが完成しました。ひとまずセーブしておきましょう。

```
save "dmail2.bas" ⏎
```

くれぐれもファイル名を前節と同じにしないように注意してください。数字が1文字違うだけですから……。

プログラムの実行

ではプログラムを実行し、コード番号101の人のデータを次のように修正してみましょう。

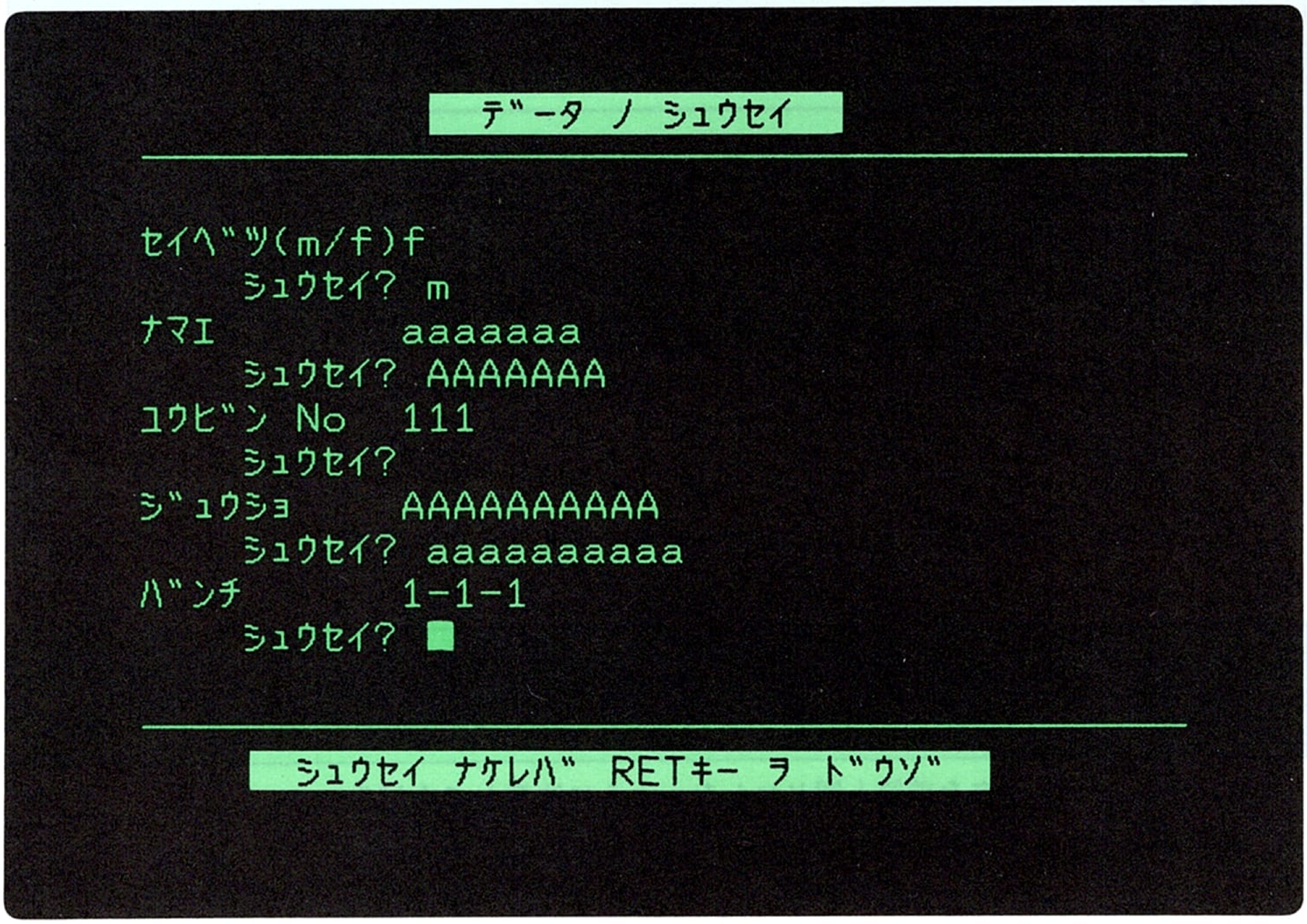
性別　m ──→ f
氏名　AAAAAAA ──→ aaaaaaa
郵便番号は無修正
住所　aaaaaaaaaa ──→ AAAAAAAAAA
番地は無修正

f・5 キーでプログラムをRUNさせてください。

するとコード入力の初期画面になりますから101をキーインし、メッセージに従って上のように変更してみてください。

4つの項目の修正データを入力後、登録するかどうかの確認メッセージに対して「Y」をキーインし、再びコード入力に戻ったら、もう一度101を入力してみてください。つまり、正しく修正データが書き込まれたかどうかを調べるのです。

次に示すのは、2回目に101を入力し、先ほどの修正データを元に戻しているところです。



番地はそのままということでリターンキーを押しますと登録するか

どうか聞いてきますから「Y」を入力しましょう。

次にふたたびコード入力になりますから、「9999」を入力して終りにします。このときメニュープログラムを実行しようとしますが、まだ作ってないので、

File not found

のエラーになりますが別にかまいません。

データ修正のプログラムリストを次に示します。

```
100 ' file name ････> dmail2.bas
110 ' 1985.2.10
120 ' ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ
130 ' save "dmail2.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
220 GOSUB *YOMU : DIM CD(COUNT) : COUNT=0
230 NIKAIME=-1 : GOSUB *YOMU
240 '
250 COLOR 4
260 LOCATE 31,0 : PRINT "  ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ  "
270 COLOR 0
280 LOCATE 20,1 : PRINT STRING$(40,149)
290 LOCATE 20,14: PRINT STRING$(40,149)
300 '
310 '-----------------------------------------------------------
320 *LOOP
330 '-----------------------------------------------------------
340 COLOR 6
350 LOCATE 40,3 : PRINT "  ｵﾜﾘ ﾉ ﾄｷﾊ 9999  "
360 COLOR 0
370 LOCATE 20,3 : INPUT "ｺｰﾄﾞ          ";CD
380 IF CD=9999 THEN GOTO *OWARI
390 GOSUB *SAGASU
400 IF P=-1 THEN GOTO *LOOP
```

```
410 IF NOT SHUUSEI THEN GOTO *LOOP
420 LOCATE 25,15 : COLOR 4
430 PRINT " ｼｭｳｾｲ ﾅｹﾚﾊﾞ RETｷｰ ｦ ﾄﾞｳｿﾞ"
440 COLOR 0
450 LOCATE 20,3
460 PRINT TAB(20) : PRINT "ｾｲﾍﾞﾂ(m/f)";SEX$(P)
470 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A1$
480 IF A1$="" THEN A1$=SEX$(P)
490 PRINT TAB(20) : PRINT "ﾅﾏｴ        ";NM$(REC)
500 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A2$
510 IF A2$="" THEN A2$=NM$(REC)
520 PRINT TAB(20) : PRINT "ﾕｳﾋﾞﾝ No   ";PN$(REC)
530 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A3$
540 IF A3$="" THEN A3$=PN$(REC)
550 PRINT TAB(20) : PRINT "ｼﾞｭｳｼｮ     ";AD$(REC)
560 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A4$
570 IF A4$="" THEN A4$=AD$(REC)
580 PRINT TAB(20) : PRINT "ﾊﾞﾝﾁ       ";BA$(REC)
590 PRINT TAB(24) : INPUT "ｼｭｳｾｲ";A5$
600 IF A5$="" THEN A5$=BA$(REC)
610 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
620 COLOR 4 : LOCATE 25,15
630 PRINT "      ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?      "
640 COLOR 0
650 I$=INKEY$ : IF I$="" THEN GOTO 650
660 IF I$="N" OR I$="n" THEN GOTO *KESU
670 IF I$="Y" OR I$="y" THEN GOTO *KAKU
680 BEEP : BEEP : GOTO 650
690 '
700 '-----------------------------------------
710 *KAKU
720 '-----------------------------------------
730 LSET SEX$(P) =A1$
740 LSET NM$(REC)=A2$
750 LSET PN$(REC)=A3$
760 LSET AD$(REC)=A4$
770 LSET BA$(REC)=A5$
780 PUT #1,SEC : PUT #2,1
790 '
800 '-----------------------------------------
810 *KESU
820 '-----------------------------------------
```

```
830 LOCATE 20,3
840 FOR I=1 TO 10
850   PRINT TAB(20);SPC(40)
860 NEXT I
870 LOCATE 20,15 : PRINT SPC(40)
880 SHUUSEI=0
890 GOTO *LOOP
900 '
910 '────────────────────────────
920 *OWARI
930 '────────────────────────────
940 CLS
950 CLOSE
960 RUN "menu1.bas"
970 '
1000 '───────────────────────────
1010 *FOPEN1
1020 '───────────────────────────
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
         6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '───────────────────────────
2010 *FOPEN2
2020 '───────────────────────────
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
2080 '
3000 '───────────────────────────
3010 *YOMU
3020 '───────────────────────────
3030 OPEN "2:indx01.dat" FOR INPUT AS #3
3040 WHILE NOT EOF(3)
3050   COUNT=COUNT+1
3060   IF NIKAIME THEN GOTO 3080
3070   INPUT #3,DUMMY : GOTO 3090
```

```
3080   INPUT #3,CD(COUNT)
3090 WEND
3100 CLOSE #3
3110 RETURN
3120 '
4000 '----------------------------------------
4010 *SAGASU
4020 '----------------------------------------
4030 P=SEARCH(CD,CD)
4040 IF P=-1 THEN GOSUB *MITOOROKU
4050 SEC=(P-1) ¥ 3+2
4060 REC=(P-1) MOD 3+1
4070 GET #1,SEC : GET #2,1
4080 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
4090 LOCATE 20,3 : PRINT SPC(40)
4100 LOCATE 20,3 : PRINT "ｼｭｳｾｲ ﾃﾞｰﾀ : ";
4110 COLOR 4 : PRINT NM$(REC) : COLOR 0
4120 LOCATE 28,15 : COLOR 6
4130 PRINT " ｼｭｳｾｲ ｼﾏｽｶ ( y/n )? "
4140 COLOR 0
4150 I$=INKEY$ : IF I$="" THEN GOTO 4150
4160 IF I$="Y" OR I$="y" THEN GOTO 4190
4170 IF I$="N" OR I$="n" THEN GOTO 4200
4180 BEEP : BEEP : GOTO 4150
4190 SHUUSEI=-1
4200 LOCATE 20,3 : PRINT SPC(40)
4210 LOCATE 20,15 : PRINT SPC(40)
4220 RETURN
4230 '
5000 '----------------------------------------
5010 *MITOOROKU
5020 '----------------------------------------
5030 BEEP : BEEP : BEEP
5040 LOCATE 20,3 : PRINT SPC(40)
5050 LOCATE 31,3 : COLOR 6
5060 PRINT "  ﾄｳﾛｸ ｼﾃｱﾘﾏｾﾝ ! " : COLOR 0
5070 FOR I=1 TO 5000 : NEXT I
5080 LOCATE 20,3 : PRINT SPC(40)
5090 RETURN 4220
```

1·7 宛て名の印刷

この節では、私たちがランダムファイルに登録した人たちの宛て名をディスプレイに表示、あるいはプリンタへ出力してみたいと思います。プリンタをお持ちでない方もいらっしゃると思いますので、どちらの方にも使えるように兼用のプログラムにするつもりです。

印刷またはディスプレイ表示までの手順

それではいつものように、このプログラムを実行すると画面がどのように変化し、最終的な結果を得られるのか、順を追って説明していきましょう。

まずプログラムをRUNすると、図1のようなメニュー画面が現れます。

1　**全部のデータ……登録してある人全員の宛て名印刷を行います。**
2　**男性のデータ……男性だけの宛て名印刷を行います。**
3　**女性のデータ……女性だけの宛て名印刷を行います。**

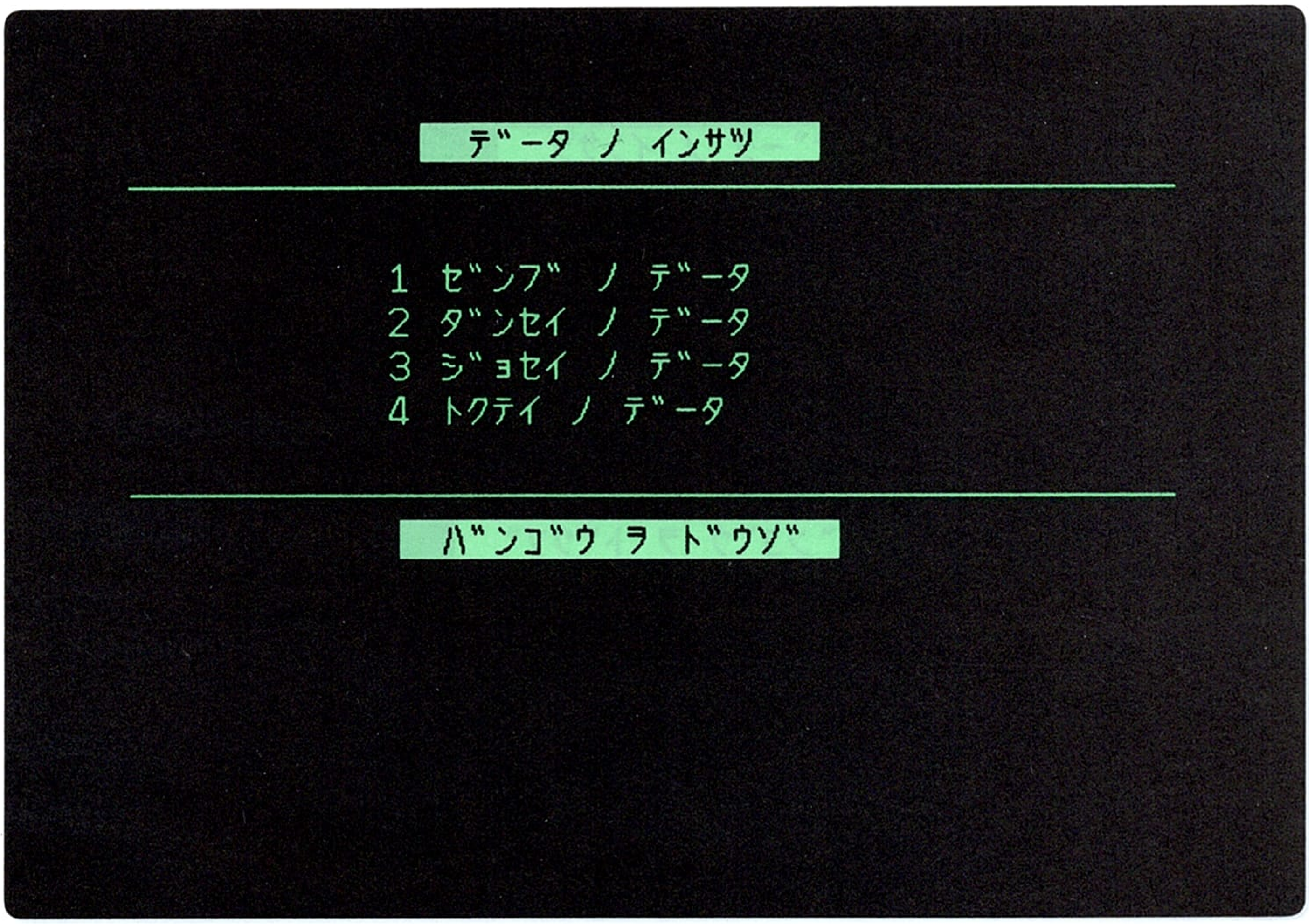


図1 ● 印刷メニュー画面

4　特定のデータ……コード入力により宛て名印刷をする人を指定します

メニュー番号の入力が終ると次は、結果をどこに出力するのか、**出力デバイス**（装置）の指定画面です。画面は**図2**のようになりますから、いずれかを選んでください。

プリンタをお持ちでない方もいると思いますので、ディスプレイへも出力できるようにしました。

次に、たとえば、メニュー番号4の特定のデータを選択すると、出力デバイスを指定したあと**図3**のように**コード入力画面**になります。いつもと同じように終りのときは「9999」を入力しリターンキーを押すことにします。

コードを入力すると、登録データの中に該当コードがあるかどうか探し出し、もしあれば**図4**のように氏名を反転表示させて、印刷データとするかどうか確認してきます。反対に、もし該当データが見つからなければ、**図5**のように**「トウロク　シテアリマセン！」**のエラーメッセージを表示します。

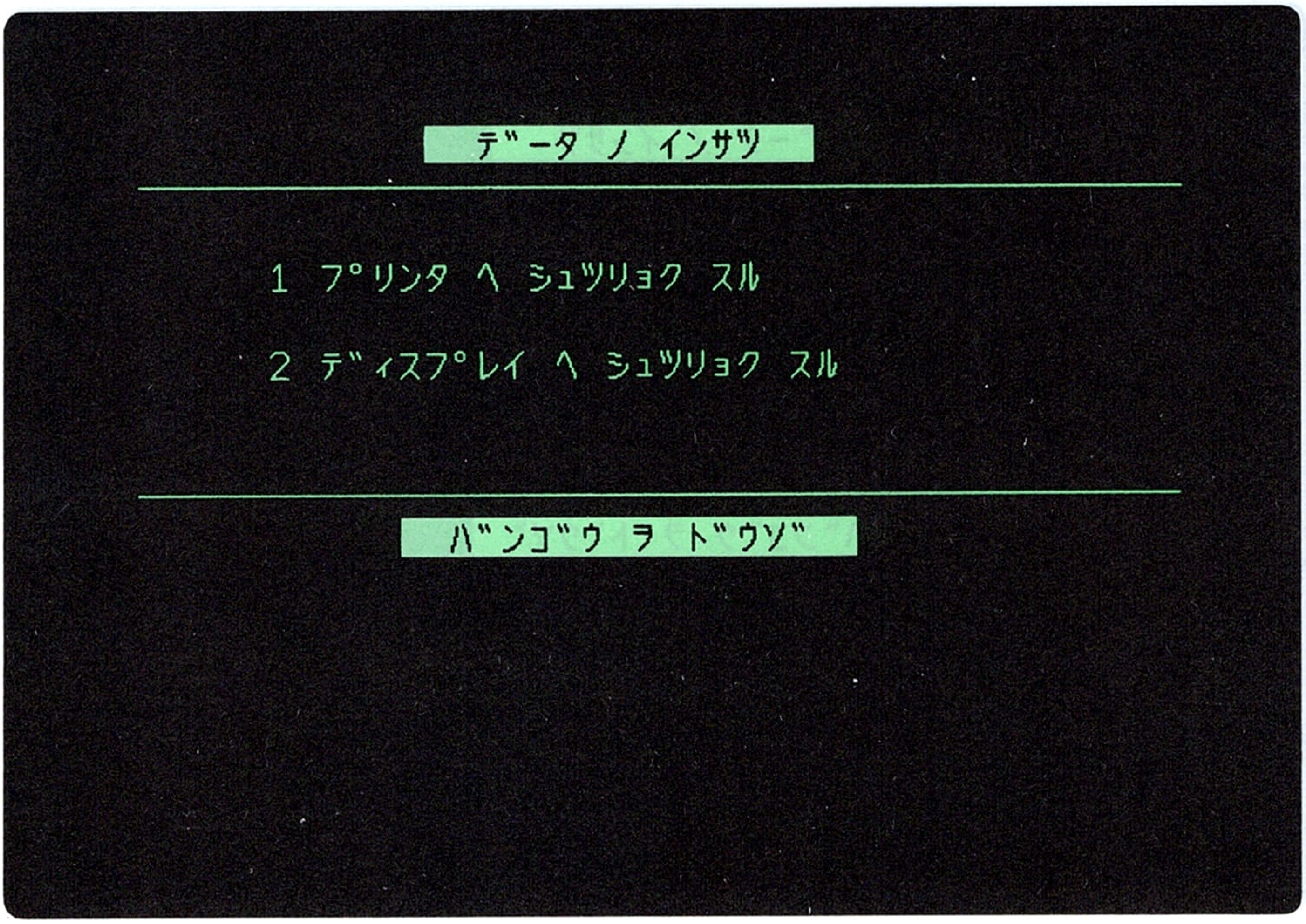


図2●出力デバイスの指定

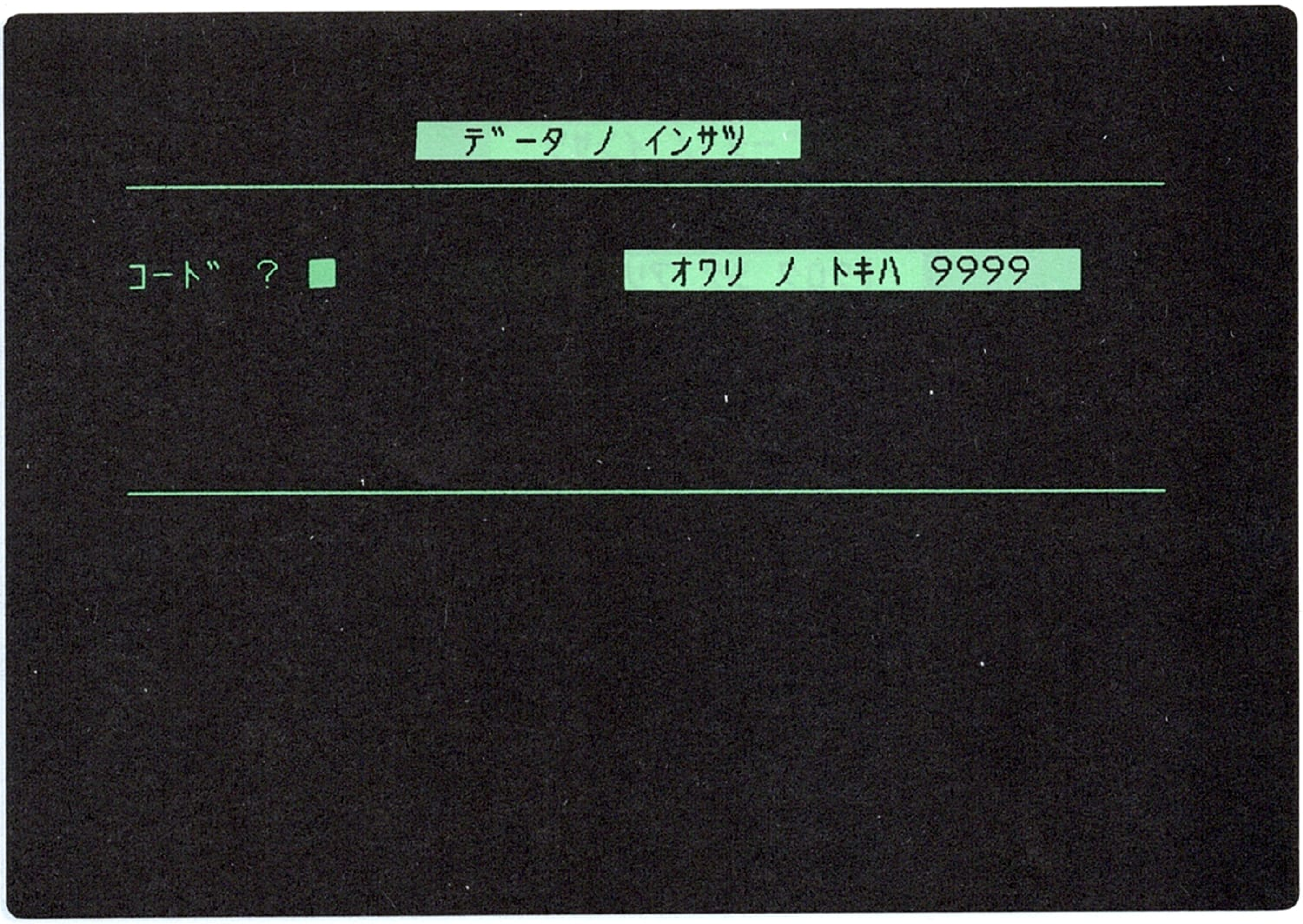


図3 コード入力画面

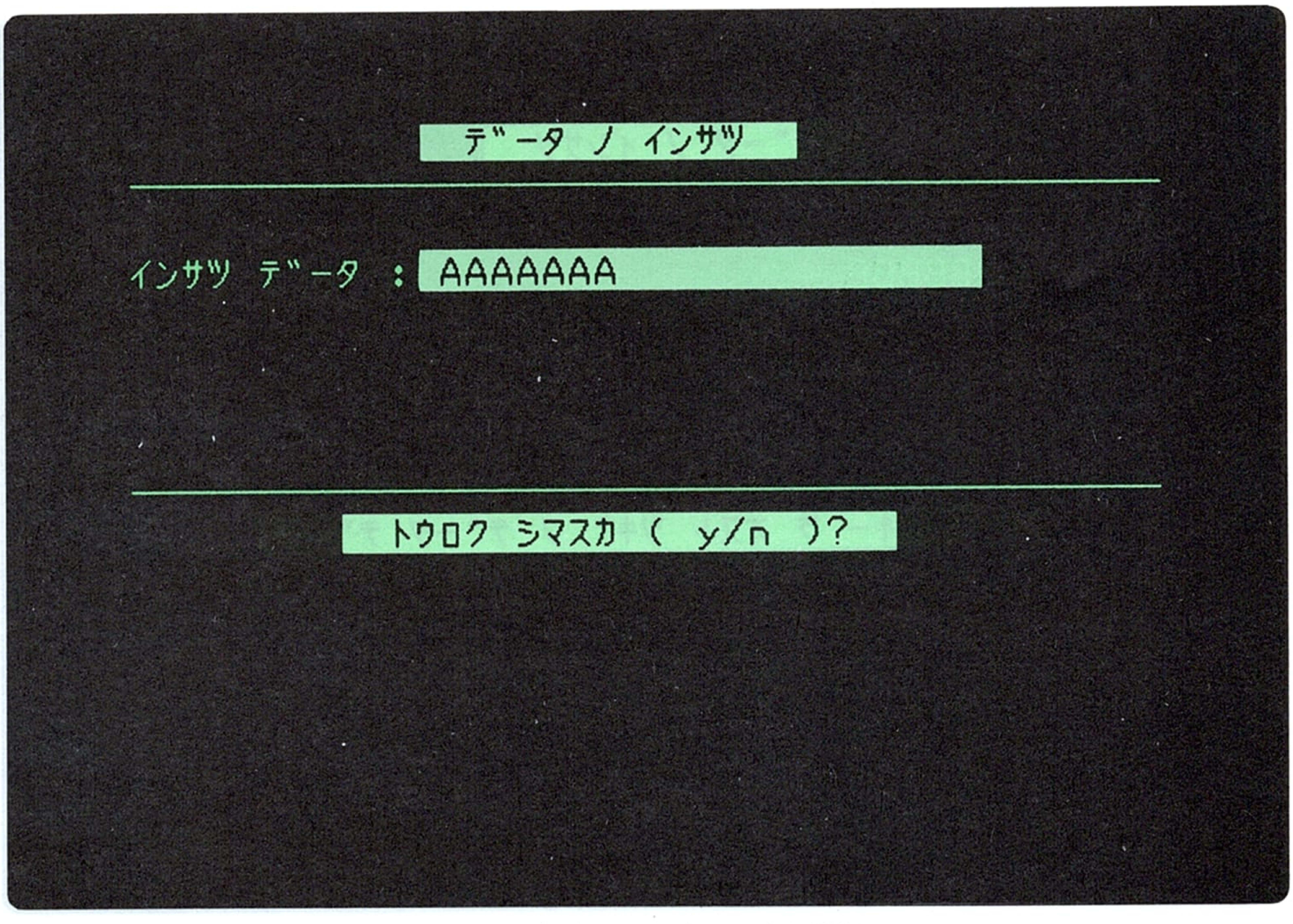


図4 登録するかどうかの確認画面

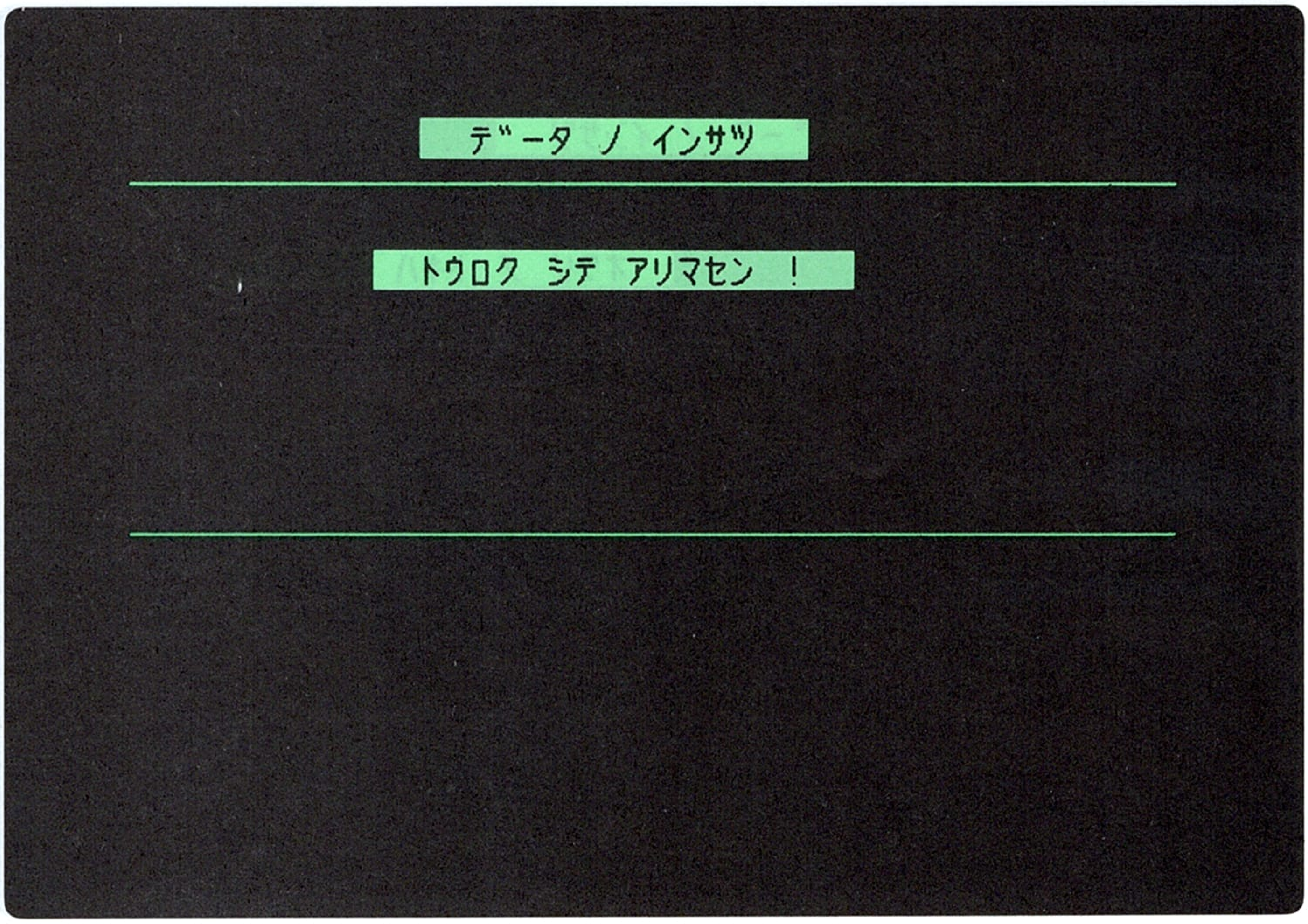


図5 ●エラーメッセージ

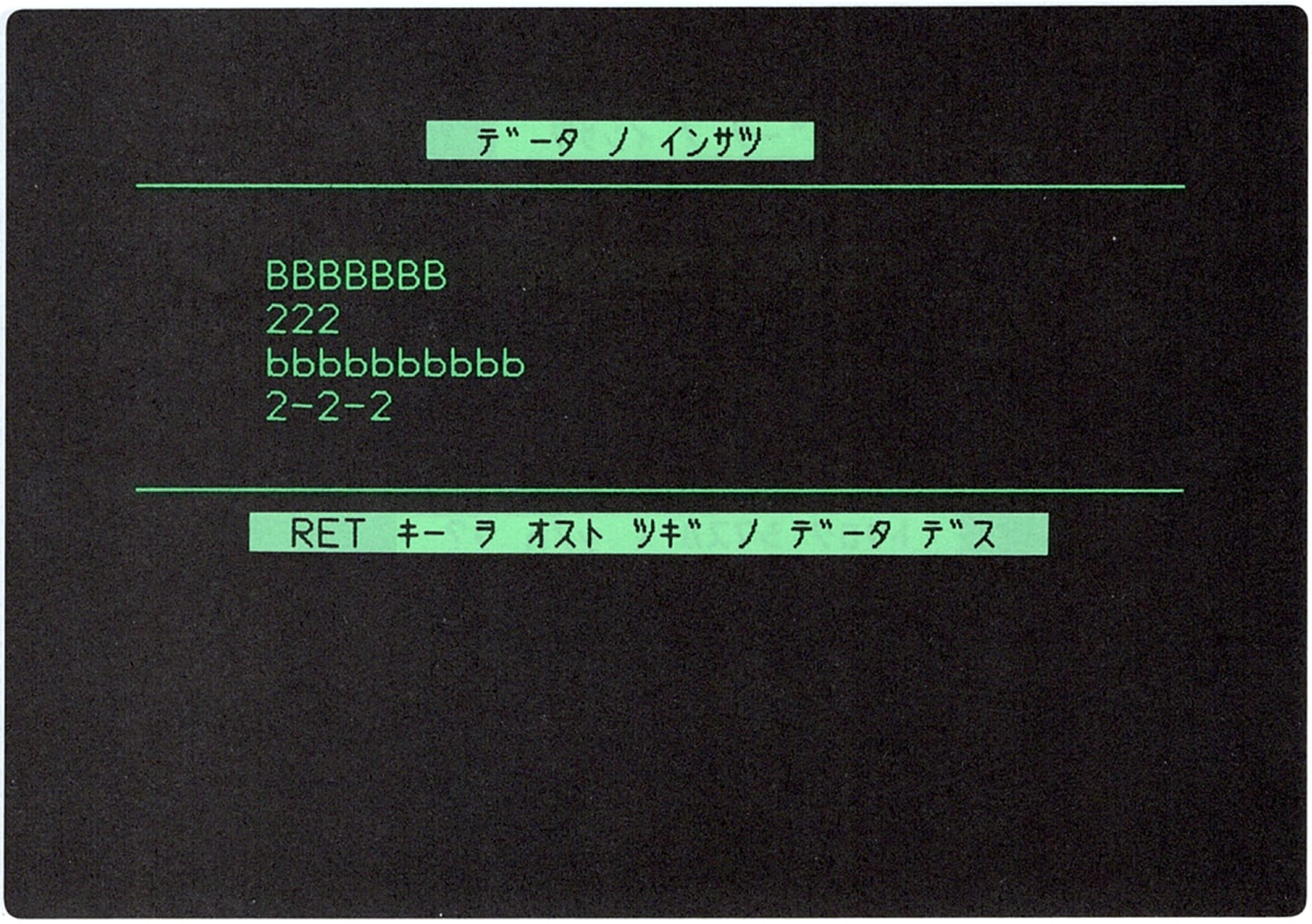


図6 ●ディスプレイへの出力

ディスプレイへの出力

図6は**ディスプレイへのデータ出力**の様子です。リターンキーを押すと次々にデータが画面に表示され、すべての表示が終るとメニュープログラムへ戻ります。

プリンタへの出力

下に示すのは、**コード入力によるプリンタへの出力例**です。この場合にはディスプレイへの出力と違って、郵便番号、住所、番地、氏名の順で出力します。**図6**を見るとわかるようにCRT（ディスプレイ）への出力は、氏名が一番最初でしたね。

```
111
aaaaaaaaaa
                    1-1-1
          AAAAAAA
                         サマ


555
eeeeeeeeee
                    5-5-5
          EEEEEEE
                         サマ


888
hhhhhhhhhh
                    8-8-8
          HHHHHHH
                         サマ
```

初期設定からランダムファイルオープンまで

以上の画面説明によるプログラムの流れを頭においてプログラムにかかりますが、まず最初はランダムファイルのオープンまでです。

次のプログラムを入力してください。

```
100 ' file name ・・・> dmail3.bas
110 ' 1985.2.10
120 ' ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ
130 ' save "dmail3.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
```

この部分は、ファイル名、処理内容などを除けば前節とまったく同じです。ファイルオープンのサブルーチンも同じですが、めんどうがらずに入力してしまいましょう。

```
1000 '────────────────────────────
1010 *FOPEN1
1020 '────────────────────────────
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
         6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '────────────────────────────
2010 *FOPEN2
2020 '────────────────────────────
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
2080 '
```

配列宣言とデータ格納

本節でのプログラムでは**コード入力によるデータサーチ**がありますので、シーケンシャルファイルのコードを一度配列の中に格納しておかなければなりません。

```
220 GOSUB *YOMU
230 DIM CD(COUNT),TR(COUNT) : COUNT=0
240 NIKAIME=-1 : GOSUB *YOMU
```

220行から次のサブルーチンへジャンプします。

```
3000 '―――――――――――――――――――――――――――――
3010 *YOMU
3020 '―――――――――――――――――――――――――――――
3030 OPEN "2:indx01.dat" FOR INPUT AS #3
3040 WHILE NOT EOF(3)
3050  COUNT=COUNT+1
3060  IF NIKAIME THEN GOTO 3080 ←――2回目なら3080行へ
3070  INPUT #3,DUMMY : GOTO 3090 ←――1回目ならカラ読み
3080  INPUT #3,CD(COUNT) ←――コードを配列に格納
3090 WEND
3100 CLOSE #3
3110 RETURN
3120 '
```

最初のジャンプでは**変数NIKAIMEの内容は**

NIKAIME=0

で「**偽**」ですから3060行の条件文にはひっかからず3070行を実行することになります。つまりデータの読み飛ばしを行い、**データ数をカウントする**（数える）わけです。

そして、このサブルーチンから戻って、変数COUNTの内容を添字として配列CD、TRを宣言します。**配列CDはコード格納用**ですが、**配列TR**は印刷するデータの登録順位を一時退避させておくためのものです。

配列宣言ができたらもう一度サブルーチンへジャンプし、今度はコードを配列CDの中に格納します。

ジャンプする前に**変数COUNTの中身をゼロクリア**しておくことを忘れないようにしましょう。

〈注〉変数NIKAIMEは、「2回目」をローマ字になおしただけです。変数には何を使ってもかまいませんが、こうしておくと条件文に読みやすさがでてきます。変数NIKAIMEの内容が真であれば3060行は「もし2回目なら〜」と読めます。しかし、サブルーチンへの2回目のジャンプかどうかを判定するフラグにFなどを使いますと「もしFが真なら〜」としか読むことができません。

メニュー選択サブルーチン

以上で基本的な準備は終りましたので次は、どのデータを出力するのかメニュー画面から選ばせる部分です。

```
250 GOSUB *MENU
```

そして次のサブルーチンを入力してください。

```
4000 '──────────────────────────────
4010 *MENU
4020 '──────────────────────────────
4030 COLOR 4
4040 LOCATE 30,0 : PRINT "   ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ    "
4050 COLOR 0
4060 LOCATE 20,1 : PRINT STRING$(40,149)
4070 LOCATE 20,8 : PRINT STRING$(40,149)
4080 LOCATE 30,3 : PRINT "1 ｾﾞﾝﾌﾞﾉ ﾃﾞｰﾀ"
4090 LOCATE 30,4 : PRINT "2 ﾀﾞﾝｾｲ ﾉ ﾃﾞｰﾀ"
4100 LOCATE 30,5 : PRINT "3 ｼﾞｮｾｲ ﾉ ﾃﾞｰﾀ"
4110 LOCATE 30,6 : PRINT "4 ﾄｸﾃｲ ﾉ ﾃﾞｰﾀ"
4120 FOR I=1 TO 300 : BEEP 1 : BEEP 0 : NEXT I
4130 LOCATE 30,9 : COLOR 6
4140 PRINT "  ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ   " : COLOR 0
4150 A$=INKEY$ : IF A$="" THEN GOTO 4150
4160 IF A$<"1" OR A$>"4" THEN BEEP : GOTO 4150
4170 LOCATE 0,3
4180 FOR I=1 TO 4
4190   PRINT TAB(20);SPC(40)
4200 NEXT I
4210 RETURN
4220 '
```

この部分を実行すると73ページに示すようなメニュー画面が表示されます。

4160行は**キー入力ミスに対するフォロー**です。1 ～ 4 以外のキーが押されると**ビープ音**が鳴ります。

4180～4200行は、このサブルーチンから帰る前にメニューを消去するための**スペース出力**です。

あとはだいたいいつもと同じパターンですからわかると思います。

なお、この部分が正常に動作するかどうか試してみたい方は、4205

行とでもして、

4205 END

とし、ホームクリアキーで画面をクリアしてからダイレクトモードで

GOTO 4010 ⏎

を行なってください。

1～4以外のメニュー番号を入力してみてください。ビープ音は鳴りましたか？

確認が終ったら、

4205 ⏎

として、4205行を削除しておきましょう。

プリンタへ？ それともディスプレイへ？

次は、宛て名データをプリンタへ出力するのか、あるいはディスプレイへ出力するのか確認しておく必要があります。

```
260 GOSUB *KAKUNIN
```

そして次のサブルーチンをキーインします。

```
5000 '————————————————————————
5010 *KAKUNIN
5020 '————————————————————————
5030 LOCATE 27,3
5040 PRINT "1 プリンタ ヘ シュツリョク スル"
5050 LOCATE 27,5
5060 PRINT "2 ディスプレイ ヘ シュツリョク スル"
5070 FOR I=1 TO 300 : BEEP 1 : BEEP 0 : NEXT I
5080 B$=INKEY$ : IF B$="" THEN GOTO 5080
5090 IF B$<"1" OR B$>"2" THEN BEEP : GOTO 5080
5100 IF B$="1" THEN INSATSU=-1
5110 LOCATE 20,3 : PRINT SPC(40)
5120 LOCATE 20,5 : PRINT SPC(40)
5130 LOCATE 25,9 : PRINT SPC(40)
5140 RETURN
5150 '
```

プログラムにむずかしいところはないと思いますが1つだけ、もしメニュー番号1の

プリンタ　へ　シュツリョク　スル

を選んだ場合、5100行で、

INSATSU=−1

というように変数INSATSUの内容を**「真」**としていますが、これはIF文によって**「プリンタへ出力させるのかどうか」**を判定するときに使います。

メニュー別の処理

さて、出力デバイスの選択が終ったら次は選択メニュー別の処理を行わなければなりません。次のプログラムを入力してください。

```
270 IF A$="1" THEN GOTO *SHUTSURYOKU
280 IF A$="2" THEN S1$="m" : S2$="M" : COUNT=0
             : GOSUB *CHECK : GOTO *SHUTSURYOKU
290 IF A$="3" THEN S1$="f" : S2$="F" : COUNT=0
             : GOSUB *CHECK : GOTO *SHUTSURYOKU
300 COUNT=0
310 '
```

〈注〉上のリストでラベル名がかなり長くなっていますが、スペルを間違わないようにしてください。少し違っただけでも正常に動作しません。

270行はメニュー番号1の

全部のデータ

を選んだ場合で、この場合はランダムファイルに書き込まれている順にすべて出力すればよいので、何の細工もせず出力ルーチンへ飛ばしてしまいます。

280行、290行は

男性のデータ

女性のデータ

を選んだ場合の処理ですが、この場合はランダムファイルの第1セクタのフラグをチェックして男性か女性かの判定をしなければなりませんので、そのためのサブルーチンへ一度ジャンプさせます。そして戻ってきてから出力ルーチンへ飛ばします。

なお、変数COUNTはカウンタ用にチェックルーチンで使いますの

でゼロクリアしておきます。

また、たとえば280行でＳ１＄、Ｓ２＄に小文字と大文字のエムを代入していますが、これは、データ書き込み時に、誤って大文字を入力していることも考えられますので、そのための処置です。

300行以降はメニュー番号４の

特定のデータ

を選んだ場合の処理になります。

男性データか女性データかのチェック

メニュー番号２または３を選択したときは、このサブルーチンへ飛んできます。

```
6000 '―――――――――――――――――――――――
6010 *CHECK
6020 '―――――――――――――――――――――――
6030 GET #2,1
6040 FOR I=1 TO 256
6050   IF SEX$(I)=S1$ OR SEX$(I)=S2$
         THEN COUNT=COUNT+1 : TR(COUNT)=I
6060 NEXT I
6070 RETURN
6080 '
```

すでにオープンされているランダムファイルの第１セクタをGETし、Ｓ１＄、Ｓ２＄の中にしまわれているフラグと比較して、もし該当するものであればカウンタを１つ上げて、そのときのループ変数Ｉの値(これが何番目のレコードかを示しています）を配列**TR**に格納しています。

たとえば本書で私たちはこれまで10人の宛て名を登録してきましたが、そのうちわけは次のとおりです。

1	2	3	4	5	6	7	8	9	10	―― ❶
m	m	f	f	m	f	m	m	f	f	―― ❷

ですから、もしメニュー画面で「女性のデータ」を選択したとしたならば、Ｓ１＄、Ｓ２＄にはｆかＦが代入されてこのサブルーチンへ入りますから、これと❷のフラグを比較していきます。

そしてI＝3のループで初めてfに会いますから、そのときカウンタCOUNTは

COUNT=COUNT+1=0+1=1

となり、

TR(1)=3

というように、配列TRの1番目の要素に3が代入されます。

同様に、

TR(2)=4, TR(3)=6, TR(4)=9, TR(5)=10

となります。

つまり、配列TRの中に、印刷すべきデータの登録順位が格納されたことになりますね。ですから、この値によって、

GETすべきセクタ番号
そのセクタ内でのレコード番号

を計算し、ランダムファイルから読み出していけばよいのです。

コード入力による検索と登録

コード入力

270〜290行のIF文にひっかからない場合は、このルーチンへ落ちてきます。

```
320 '―――――――――――――――――――――――――
330 *LOOP
340 '―――――――――――――――――――――――――
350 COLOR 6
360 LOCATE 40,3 : PRINT " ｵﾜﾘ ﾉ ﾄｷﾊ 9999 "
370 COLOR 0
380 LOCATE 20,3 : INPUT "ｺｰﾄﾞ";CD
390 IF CD=9999 THEN GOTO 420
400 GOSUB *SAGASU
410 GOTO *LOOP
420 LOCATE 20,3 : PRINT SPC(40)
430 '
```

75ページの図3のコード入力画面を思い出してください。この部分を実行したときの様子がわかると思います。

コードに対して「9999」が入力されるとこのルーチンから抜け出て印刷ルーチンへ飛びます。

それ以外の場合は**データサーチのサブルーチン**へジャンプします。 データサーチ

```
6090 '―――――――――――――――――――――――――――――――――――――――――――――
6100 *SAGASU
6110 '―――――――――――――――――――――――――――――――――――――――――――――
6120 P=SEARCH(CD,CD)
6130 IF P=-1 THEN GOSUB *MITOOROKU
6140 SEC=(P-1) ¥ 3+2
6150 REC=(P-1) MOD 3+1
6160 GET #1,SEC
6170 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
6180 LOCATE 20,3 : PRINT SPC(40)
6190 LOCATE 20,3 : PRINT "ｲﾝｻﾂ ﾃﾞｰﾀ : ";
6200 COLOR 4 : PRINT NM$(REC) : COLOR 0
6210 LOCATE 28,9 : COLOR 6
6220 PRINT "ﾄｳﾛｸ ｼﾏｽｶ ( y/n )? "
6230 COLOR 0
6240 I$=INKEY$ : IF I$="" THEN GOTO 6240
6250 IF I$="Y" OR I$="y" THEN GOTO 6280
6260 IF I$="N" OR I$="n" THEN GOTO 6290
6270 BEEP : BEEP : GOTO 6240
6280 COUNT=COUNT+1 : TR(COUNT)=P
6290 LOCATE 20,3 : PRINT SPC(40)
6300 LOCATE 20,9 : PRINT SPC(40)
6310 RETURN
6320 '
```

SEARCH関数を使って、入力されたコードが配列CDの中にあるかどうか探します。 SEARCH関数

見つからなければ値として「－１」を返しますから、この場合はエラーメッセージ表示のサブルーチンへ飛ばします。このサブルーチンはあとで示します。

見つかった場合はその値を元にしてセクタ番号、そのセクタ内でのレコード番号を計算し、まず該当セクタをGET文でバッファ上に取り出します。そして、6200行で氏名を画面に表示し、印刷のために登録するかどうか確認をうながします。

もし登録する場合には6280行でカウンタを１つ上げて、それを添字として配列TRに**「登録順位」**をしまいます。たとえば、コード番号501の人なら登録順位は５番目ですからTR(1)に「５」が格納されます。

登録するしないにかかわらず、画面下のオペレーションメッセージ、確認のための表示データを消してRETURNです。

該当コードなし

次に示すのは**該当コードがない場合のサブルーチン**です。

```
7000 '────────────────────────────
7010 *MITOOROKU
7020 '────────────────────────────
7030 BEEP : BEEP : BEEP
7040 LOCATE 20,3 : PRINT SPC(40)
7050 LOCATE 28,3 : COLOR 6
7060 PRINT "  ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ !  " : COLOR 0
7070 FOR I=1 TO 5000 : NEXT I
7080 LOCATE 20,3 : PRINT SPC(40)
7090 RETURN 6310
```

このサブルーチンから戻る場合、その戻り先をRETURN文で6310行に指定していることに注意してください。

出力ルーチン

全部のデータ、男性、女性のデータ、特定のデータの出力、いずれの場合も次のルーチンへ飛んできます。

データ出力

```
440 '────────────────────────────
450 *SHUTSURYOKU
460 '────────────────────────────
470 IF INSATSU THEN GOTO *PRT
480 GOTO *CRT
490 '
```

470行の

IF INSATSU THEN～

はわかりますね。82ページで説明したように、**「プリンタへ出力する」**場合を選びますと変数INSATSUの内容は－1で**「真」**となっていますから、THEN文以下が実行され、**プリンタへの出力ルーチン**

***PRT**

へジャンプすることになります。

```
500 '────────────────────────────
510 *PRT
520 '────────────────────────────
530 FOR I=1 TO COUNT                          ←─ 全データを印刷するとき
540   IF TR(I)=0 THEN P=I : GOTO 560
550   P=TR(I)                                 ←─ 登録順位の代入
560   SEC=(P-1) ¥ 3+2                         ←─ セクタ番号の計算
570   REC=(P-1) MOD 3+1                       ←─ セクタ内のレコード番号の計算
580   GET #1,SEC                              ←─ セクタの内容をバッファ上へ
590   LPRINT PN$(REC)                         ┐
600   LPRINT AD$(REC)                         │
610   LPRINT TAB(10);BA$(REC)                 ├─ プリンタへ出力
620   LPRINT TAB(5);NM$(REC)                  │
630   LPRINT TAB(15);"ｻﾏ"                     ┘
640   FOR J=1 TO 5 : LPRINT : NEXT J          ←─ 5行分だけ紙送りをする
650 NEXT I
660 GOTO *OWARI
670 '
680 '────────────────────────────
690 *CRT
700 '────────────────────────────
710 COLOR 6 : LOCATE 25,9
720 PRINT " RET ｷｰ ｦ ｵｽﾄ ﾂｷﾞ ﾉ ﾃﾞｰﾀ ﾃﾞｽ "
730 COLOR 0
740 FOR I=1 TO COUNT
750   IF TR(I)=0 THEN P=I : GOTO 770
760   P=TR(I)
770   SEC=(P-1) ¥ 3+2
780   REC=(P-1) MOD 3+1
790   GET #1,SEC
800   LOCATE 25,3 : PRINT NM$(REC)
810   LOCATE 25,4 : PRINT PN$(REC)
820   LOCATE 25,5 : PRINT AD$(REC)
830   LOCATE 25,6 : PRINT BA$(REC)
840   I$=INKEY$ : IF I$="" THEN GOTO 840
850 NEXT I
860 '
```

それ以下の場合は、CRTディスプレイへの出力ルーチン

＊CRT

へジャンプします。いずれの場合もリストを上に示します。

コメントを入れておきましたので、ご理解いただけると思いますが、540行は、**すべてのデータを印刷する場合、配列TRは使っていませんでしたので、その内容はすべて0**ですね。ですからこの場合には、ループ変数Iの値を該当セクタおよびセクタ内でのレコード番号を算出する変数Pに代入しています。

一方その他の場合は、**配列TRの中に、書き込んであるデータのトップを1番とする登録順位が入っていますから、これを同じくPに代入**します。

データの書き込まれているセクタおよびセクタ内でのレコード番号の計算は、前に勉強しましたのでわかると思います。560、570行がそうですね。

以上、プリンタへ出力する場合について説明しましたが、CRTディスプレイへの出力も理屈はまったく同じです。

終りの処理

データ出力が終ったら、**オープンしているファイルをクローズ**し、メニュープログラムを実行します。

次のプログラムを入力してしまいましょう。

```
870 '―――――――――――――――――――――――
880 *OWARI
890 '―――――――――――――――――――――――
900 CLS
910 CLOSE
920 RUN "menu1.bas"
930 '
```

これで、宛て名印刷のプログラムは終りです。とりあえずセーブしておきましょう。

```
save "dmail3.bas" ⏎
```

セーブがすんだらプログラムを実行して、いろいろなケースを試してみてください。

次ページに本節でのプログラム総リストを示します。

```
100 '  file name ････> dmail3.bas
110 '  1985.2.10
120 '  ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ
130 '  save "dmail3.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 DIM NM$(3),PN$(3),AD$(3),BA$(3)
180 DIM SEX$(256)
190 '
200 GOSUB *FOPEN1
210 GOSUB *FOPEN2
220 GOSUB *YOMU
230 DIM CD(COUNT),TR(COUNT) : COUNT=0
240 NIKAIME=-1 : GOSUB *YOMU
250 GOSUB *MENU
260 GOSUB *KAKUNIN
270 IF A$="1" THEN GOTO *SHUTSURYOKU
280 IF A$="2" THEN S1$="m" : S2$="M" : COUNT=0
             : GOSUB *CHECK : GOTO *SHUTSURYOKU
290 IF A$="3" THEN S1$="f" : S2$="F" : COUNT=0
             : GOSUB *CHECK : GOTO *SHUTSURYOKU
300 COUNT=0
310 '
320 '----------------------------------------
330 *LOOP
340 '----------------------------------------
350 COLOR 6
360 LOCATE 40,3 : PRINT " ｵﾜﾘ ﾉ ﾄｷﾊ 9999 "
370 COLOR 0
380 LOCATE 20,3 : INPUT "ｺｰﾄﾞ ";CD
390 IF CD=9999 THEN GOTO 420
400 GOSUB *SAGASU
410 GOTO *LOOP
420 LOCATE 20,3 : PRINT SPC(40)
430 '
440 '----------------------------------------
450 *SHUTSURYOKU
460 '----------------------------------------
470 IF INSATSU THEN GOTO *PRT
480 GOTO *CRT
490 '
```

```
500 '--------------------------------------------------
510 *PRT
520 '--------------------------------------------------
530 FOR I=1 TO COUNT
540   IF TR(I)=0 THEN P=I : GOTO 560
550   P=TR(I)
560   SEC=(P-1) ¥ 3+2
570   REC=(P-1) MOD 3+1
580   GET #1,SEC
590   LPRINT PN$(REC)
600   LPRINT AD$(REC)
610   LPRINT TAB(10);BA$(REC)
620   LPRINT TAB(5);NM$(REC)
630   LPRINT TAB(15);"ｻﾏ"
640   FOR J=1 TO 5 : LPRINT : NEXT J
650 NEXT I
660 GOTO *OWARI
670 '
680 '--------------------------------------------------
690 *CRT
700 '--------------------------------------------------
710 COLOR 6 : LOCATE 25,9
720 PRINT " RET ｷｰ ｦ ｵｽﾄ ﾂｷﾞ ﾉ ﾃﾞｰﾀ ﾃﾞｽ "
730 COLOR 0
740 FOR I=1 TO COUNT
750   IF TR(I)=0 THEN P=I : GOTO 770
760   P=TR(I)
770   SEC=(P-1) ¥ 3+2
780   REC=(P-1) MOD 3+1
790   GET #1,SEC
800   LOCATE 25,3 : PRINT NM$(REC)
810   LOCATE 25,4 : PRINT PN$(REC)
820   LOCATE 25,5 : PRINT AD$(REC)
830   LOCATE 25,6 : PRINT BA$(REC)
840   I$=INKEY$ : IF I$="" THEN GOTO 840
850 NEXT I
860 '
870 '--------------------------------------------------
880 *OWARI
890 '--------------------------------------------------
900 CLS
910 CLOSE
```

```
920 RUN "menu1.bas"
930 '
1000 '――――――――――――――――――――――――――――――
1010 *FOPEN1
1020 '――――――――――――――――――――――――――――――
1030 OPEN "2:mail01.dat" AS #1
1040 FOR I=1 TO 3
1050   FIELD #1,85*(I-1) AS D$,20 AS NM$(I),
        6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '――――――――――――――――――――――――――――――
2010 *FOPEN2
2020 '――――――――――――――――――――――――――――――
2030 OPEN "2:mail01.dat" AS #2
2040 FOR I=1 TO 256
2050   FIELD #2,(I-1) AS D$,1 AS SEX$(I)
2060 NEXT I
2070 RETURN
2080 '
3000 '――――――――――――――――――――――――――――――
3010 *YOMU
3020 '――――――――――――――――――――――――――――――
3030 OPEN "2:indx01.dat" FOR INPUT AS #3
3040 WHILE NOT EOF(3)
3050   COUNT=COUNT+1
3060   IF NIKAIME THEN GOTO 3080
3070   INPUT #3,DUMMY : GOTO 3090
3080   INPUT #3,CD(COUNT)
3090 WEND
3100 CLOSE #3
3110 RETURN
3120 '
4000 '――――――――――――――――――――――――――――――
4010 *MENU
4020 '――――――――――――――――――――――――――――――
4030 COLOR 4
4040 LOCATE 30,0 : PRINT "    ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ    "
4050 COLOR 0
4060 LOCATE 20,1 : PRINT STRING$(40,149)
4070 LOCATE 20,8 : PRINT STRING$(40,149)
```

```
4080 LOCATE 30,3 : PRINT "1 ｾﾞﾝﾌﾞ ﾉ ﾃﾞｰﾀ"
4090 LOCATE 30,4 : PRINT "2 ﾀﾞﾝｾｲ ﾉ ﾃﾞｰﾀ"
4100 LOCATE 30,5 : PRINT "3 ｼﾞｮｾｲ ﾉ ﾃﾞｰﾀ"
4110 LOCATE 30,6 : PRINT "4 ﾄｸﾃｲ ﾉ ﾃﾞｰﾀ"
4120 FOR I=1 TO 300 : BEEP 1 : BEEP 0 : NEXT I
4130 LOCATE 30,9 : COLOR 6
4140 PRINT "  ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ   " : COLOR 0
4150 A$=INKEY$ : IF A$="" THEN GOTO 4150
4160 IF A$<"1" OR A$>"4" THEN BEEP : GOTO 4150
4170 LOCATE 0,3
4180 FOR I=1 TO 4
4190   PRINT TAB(20);SPC(40)
4200 NEXT I
4210 RETURN
4220 '
5000 '--------------------------------------------------
5010 *KAKUNIN
5020 '--------------------------------------------------
5030 LOCATE 27,3
5040 PRINT "1 ﾌﾟﾘﾝﾀ ﾍ ｼｭﾂﾘｮｸ ｽﾙ"
5050 LOCATE 27,5
5060 PRINT "2 ﾃﾞｨｽﾌﾟﾚｲ ﾍ ｼｭﾂﾘｮｸ ｽﾙ"
5070 FOR I=1 TO 300 : BEEP 1 : BEEP 0 : NEXT I
5080 B$=INKEY$ : IF B$="" THEN GOTO 5080
5090 IF B$<"1" OR B$>"2" THEN BEEP : GOTO 5080
5100 IF B$="1" THEN INSATSU=-1
5110 LOCATE 20,3 : PRINT SPC(40)
5120 LOCATE 20,5 : PRINT SPC(40)
5130 LOCATE 25,9 : PRINT SPC(40)
5140 RETURN
5150 '
6000 '--------------------------------------------------
6010 *CHECK
6020 '--------------------------------------------------
6030 GET #2,1
6040 FOR I=1 TO 256
6050   IF SEX$(I)=S1$ OR SEX$(I)=S2$
         THEN COUNT=COUNT+1 : TR(COUNT)=I
6060 NEXT I
6070 RETURN
6080 '
6090 '--------------------------------------------------
```

```
6100 *SAGASU
6110 '────────────────────────────────────
6120 P=SEARCH(CD,CD)
6130 IF P=-1 THEN GOSUB *MITOOROKU
6140 SEC=(P-1) ¥ 3+2
6150 REC=(P-1) MOD 3+1
6160 GET #1,SEC
6170 FOR I=1 TO 200 : BEEP 1 : BEEP 0 : NEXT I
6180 LOCATE 20,3 : PRINT SPC(40)
6190 LOCATE 20,3 : PRINT "ｲﾝｻﾂ ﾃﾞｰﾀ : ";
6200 COLOR 4 : PRINT NM$(REC) : COLOR 0
6210 LOCATE 28,9 : COLOR 6
6220 PRINT "  ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?  "
6230 COLOR 0
6240 I$=INKEY$ : IF I$="" THEN GOTO 6240
6250 IF I$="Y" OR I$="y" THEN GOTO 6280
6260 IF I$="N" OR I$="n" THEN GOTO 6290
6270 BEEP : BEEP : GOTO 6240
6280 COUNT=COUNT+1 : TR(COUNT)=P
6290 LOCATE 20,3 : PRINT SPC(40)
6300 LOCATE 20,9 : PRINT SPC(40)
6310 RETURN
6320 '
7000 '────────────────────────────────────
7010 *MITOOROKU
7020 '────────────────────────────────────
7030 BEEP : BEEP : BEEP
7040 LOCATE 20,3 : PRINT SPC(40)
7050 LOCATE 29,3 : COLOR 6
7060 PRINT "  ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ !  " : COLOR 0
7070 FOR I=1 TO 5000 : NEXT I
7080 LOCATE 20,3 : PRINT SPC(40)
7090 RETURN 6310
```

1・8 通信文を印刷する

せっかく宛て名が表示できたのですから、宛て名の次に通信文を印刷してみませんか？　簡単ですからやってみましょう。

前節のプログラムの590行から640行の**宛て名印刷の次に通信文の印刷処理を入れればよい**のです。この処理はサブルーチンにすることにして、**前節のプログラムをロード**してから次のステートメントを追加します。

```
644    GOSUB *BUNSHO
```

サブルーチンでは、まずその人の名前を印刷し、改行して通信文を印刷します。

次にプログラムリストを示しますが、サンプルとして、ナツメ社発行の書籍の書店向けの案内としました。ここはみなさんなりに、適当に変えてみてください。

通信文のサンプル

```
7100 '
8000 '--------------------------------------------
8010 *BUNSHO
8020 '--------------------------------------------
8030 LPRINT " ｲﾂﾓ ｵﾋｷﾀﾃ ｲﾀﾀﾞｷ ｱﾘｶﾞﾄｳ ｺﾞｻﾞｲﾏｽ｡"
8040 LPRINT " ﾏﾀ､ｷﾃﾝ ﾏｽﾏｽ ｺﾞﾊﾝｴｲ ﾉ ｺﾄﾄ ｵﾖﾛｺﾋﾞ"
8050 LPRINT "ﾓｳｼ ｱｹﾞﾏｽ｡"
8060 LPRINT " ｻﾃ ﾍｲｼｬ ﾃﾞﾊ 4月 ｹﾞｼﾞｭﾝ､ﾂｷﾞ ﾉ ｼｮｾ"
8070 LPRINT "ｷ ｦ ﾊﾂﾊﾞｲ ｲﾀｼﾏｽﾉﾃﾞ､ｺﾞﾚﾝｾｸ ｲﾀｼﾏｽ｡"
8080 LPRINT
8090 LPRINT "       PC-8801 FR ﾃﾞｰﾀ ﾌｧｲﾙ"
8100 LPRINT
8110 LPRINT " ﾎﾝｼｮ ﾊ ﾃﾞｨｽｸﾅｲｿﾞｳ ﾉ ﾊﾟｿｺﾝ ｦ ｼﾞﾂ"
8120 LPRINT "ﾑ ﾆ ﾄﾞｳ ｶﾂﾖｳｽﾙｶ ｦ ﾜｶﾘﾔｽｸ ｶｲｾﾂｼﾀﾓ"
8130 LPRINT "ﾉ ﾃﾞ､ｶﾅﾘ ﾉ ｼﾞｭﾜｳ ｶﾞ ﾐｺﾏﾚﾏｽ｡ﾁｭｳﾓﾝ"
8140 LPRINT "ｼｮ ｦ ﾄﾞｳﾌｳ ｲﾀｼﾏｼﾀ ﾉﾃﾞ､ｾﾞﾋ ｺﾞﾊﾝﾊﾞ"
8150 LPRINT "ｲｸﾀﾞｻﾙﾖｳ ｵﾈｶﾞｲ ﾓｳｼｱｹﾞﾏｽ｡"
8160 LPRINT
8170 LPRINT "                    ﾅﾂﾒｼｬ ｴｲｷﾞｮｳﾌﾞ"
8180 FOR J=1 TO 5 : LPRINT : NEXT J
8190 RETURN
```

カタカナの通信文というのはちょっと抵抗がありますが、ここでは理屈を学ぶのが目的ですから我慢しましょう。

PC-8801MKIISRには漢字ROMが実装されていますから、カタカナではなく漢字出力も可能ですが、PC-9801Mのように日本語表示選択ができないので、1つ1つ漢字コードをひろっていかなければならず大変不便です。

なお、8180行のループ変数をIにしないでください。というのは、90ページの530～650行を見るとわかりますが、ここがすでにIのループになっています。そして、このループの中からサブルーチンへ飛んできたのですから、ループ変数は他のものに変えないと正常に動作しません。

このサブルーチンは、ごく普通の**LPRINT文**ですが、8180行のステートメントを入れ5行分の**紙送り**をしておきませんと、次の人の宛て名が続いて出力されてしまいます。

LPRINT文

紙送り

これを実行してみるとわかりますが、3人以上出力すると、2ページにまたがって印刷され、3人目の人の宛て名がミシン目の上にのってしまいます。これではちょっとまずいので、**2人分印刷したら改ページする**ようにしておきます。

改ページ

●PRINT CHR$(12)は画面をクリアして新しい画面にすること。これと同じでLPRINT CHR$(12)は印刷用紙を新しいページにすることです。

次のステートメントを追加してください。

```
642    RC=RC+1
646    IF RC=2 THEN RC=0 : LPRINT CHR$(12)
```

642行で**レコードをカウント**し、もし2人の処理がすんでいたら、646行で**改ページ処理**をしています。このときに、カウンタ用変数をゼロクリアしておくのを忘れないでください。

〈注〉変数RCは「レコードカウント」の頭文字をとったつもりです。

これでプログラムの修正は終りです。

行番号100および130のファイル名をスクリーンエディタで

dmail4.bas

と修正し、この新しいファイル名でセーブしておきましょう。

```
save "dmail4.bas" ⏎
```

さて、これでプログラムをRUNさせてみてください。

実行例を次ページに示します。

この通信文は宛て名と一緒に印刷できますので、**紙送りのLPRINT文を調整することにより、宛て名を窓つき封筒の窓のところにくるよ**

```
111
aaaaaaaaaaa
              1-1-1
     AAAAAAA
                    サマ





 イツモ  オヒキタテ  イタタ゛キ  アリカ゛トウ  コ゛サ゛イマス。
 マタ、キテン  マスマス  コ゛ハンエイ  ノ  コトト  オヨロコヒ゛
モウシ  アケ゛マス。
 サテ  ヘイシャ  テ゛ハ  4月  ケ゛シ゛ュン、ツキ゛  ノ  ショセ
キ  ヲ  ハツハ゛イ  イタシマスノテ゛、コ゛レンセク  イタシマス。

     PC-8801  FR  テ゛ータ  ファイル

 ホンショ  ハ  テ゛ィスクナイソ゛ウ  ノ  ハ゜ソコン  ヲ  シ゛ツ
ム  ニ  ト゛ウ  カツヨウスルカ  ヲ  ワカリヤスク  カイセツシタモ
ノ  テ゛、カナリ  ノ  シ゛ュワウ  カ゛  ミコマレマス。チュウモン
ショ  ヲ  ト゛ウフウ  イタシマシタ  ノテ゛、セ゛ヒ  コ゛ハンハ゛
イクタ゛サルヨウ  オネカ゛イ  モウシアケ゛マス。

                    ナツメシャ  エイキ゛ョウフ゛
```

うにし、パンフレットなどを同封すれば、十分実用的なプログラムとして活用できると思います。

また、この文例は横22字で折り返しになっていますが、まだまだふやすことができるのです。皆さん工夫してみてください。

1・9　DEF FN文を使う

DEF FNは**ディファイン・ファンクション**と読み、ユーザーによって作られた関数を定義するものです。

DEF FN

書式は次のとおりです。

DEF FN<関数名>(<パラメータリスト>)=<関数の定義式>

これをどこに使うかと申しますと、前節のプログラムの560行および570行などでGETするセクタ番号とそのセクタ内でのレコード番号を決めていますが、この部分を**DEF FN**に切り換えようというわけです。いまのままで、なんの支障もなく走るわけですから、わざわざこんなことをしなくてもよいのですが、この関数はかはり便利ですので、覚えておくとよいでしょう。

まず前節のプログラムをロードしてください。

```
load "dmail4.bas" ⏎
```

この関数をプログラム中で使うには、まず上の書式にならって関数を定義しておかなければなりません。セクタ番号とレコード番号を求めるのですから、関数名はそれぞれ**SEC**、**REC**としておきます。そして、初期設定部分で次のようなステートメントを追加しておきます。

関数の定義

```
165 DEF FNSEC(X)=(X-1) ¥ 3+2
166 DEF FNREC(X)=(X-1) MOD 3+1
```

これで関数を定義することができました。こうしておきますと、プログラムの中でいつでも呼び出すことができます。

では、560行と570行を次のように変更しますので、あらためて打ち込んでください。

```
560    SEC=FNSEC(P)
570    REC=FNREC(P)
```

そのほか、770行、780行、それに6140行、6150行も同様に変更しておきましょう。

```
770   SEC=FNSEC(P)
780   REC=FNREC(P)
```

```
6140 SEC=FNSEC(P)
6150 REC=FNREC(P)
```

このように、**カッコの中に必要なパラメータを入れますと、先ほど定義した数式にもとづいてSECおよびRECを計算してくれます。**

カッコ内のパラメータは1次元とは限らず、2次元、3次元でもかまいませんので、複数の変数からなる算式を計算する場合など、とても便利です。

それではもう一度、前節と同じファイル名でセーブしておきましょう。

```
save "dmail4.bas" ⏎
```

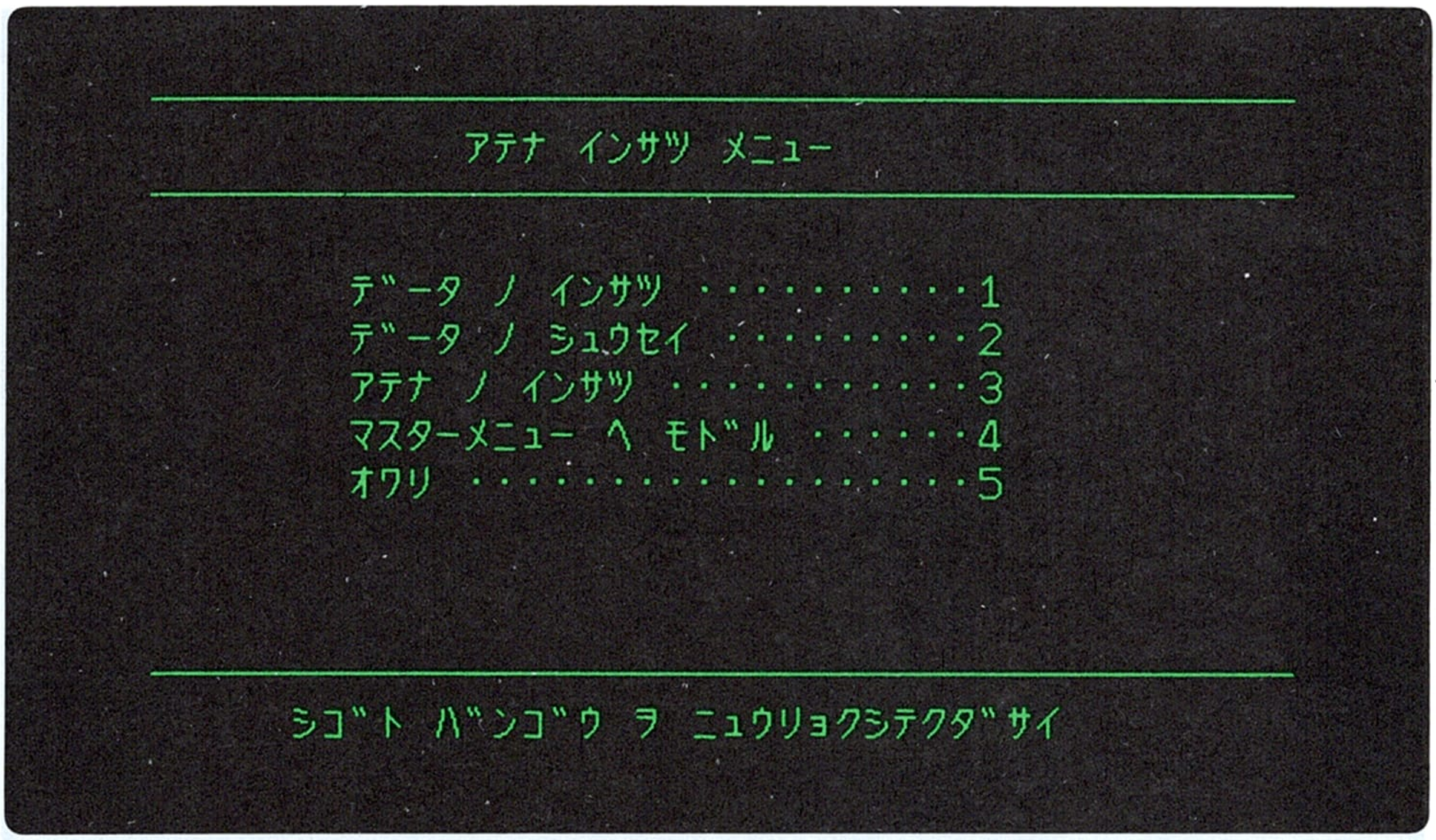
セーブできましたらプログラムを実行させてみてください。

1・10　メニュープログラム

第1章を終るに当たってメニュープログラムを作っておきましょう。

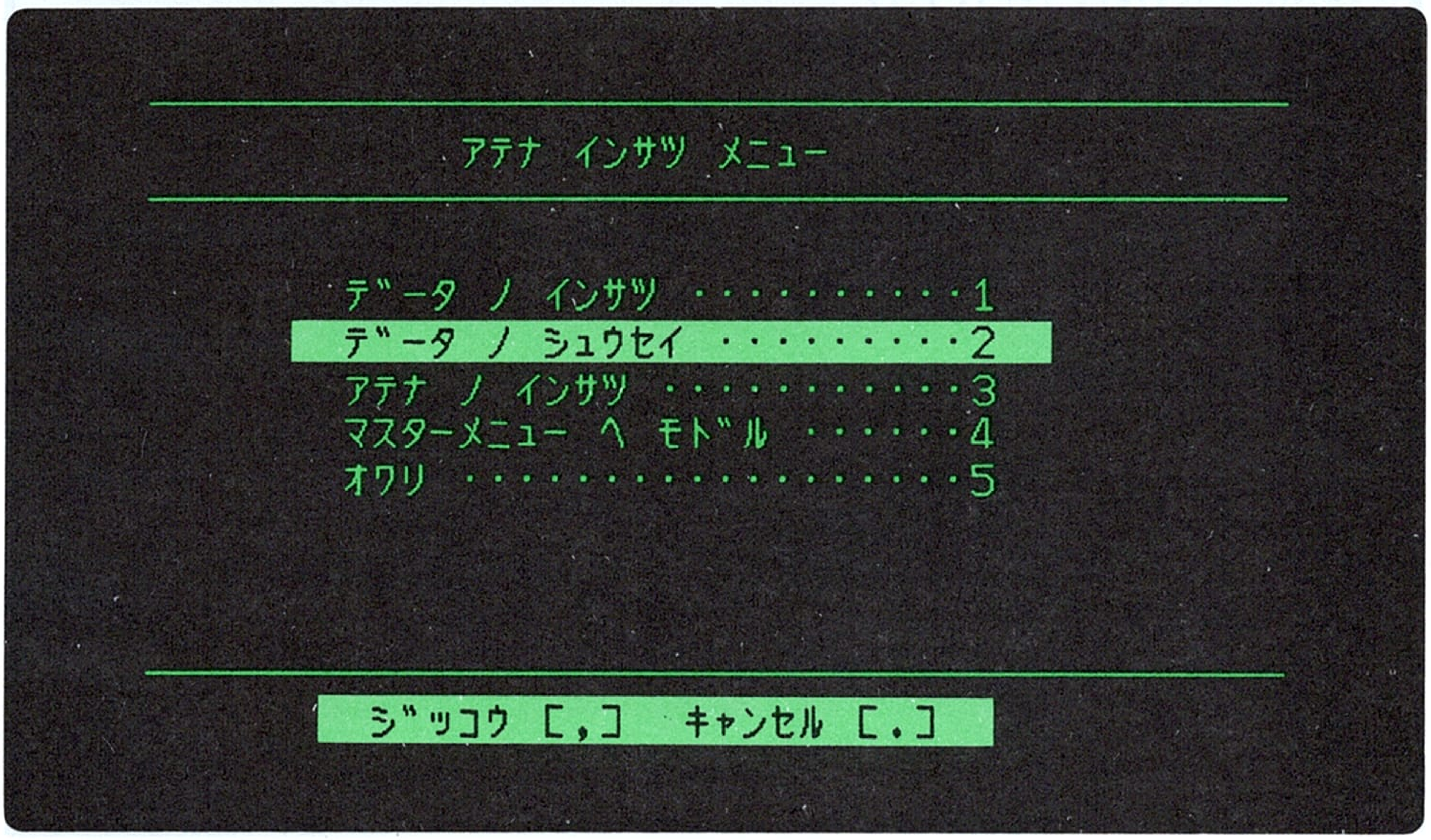
まず、このプログラムをRUNさせると次に示す画面のようにメニューが表示されます。

メニュー画面⬇



ガイドメッセージに従って好きな番号を入力すると、処理名のところが反転し、実行するかどうか聞いてきます。

確認⬇



実行するならコンマキー , 、キャンセルならピリオドキー . を押します。

プログラムリストを次に示しますが、プログラムの構造は前編のメニュープログラムとほぼ同じですので説明は省略します。

NEWコマンドでメモリをクリアし、間違わないように入力してください。

```
100 ' file name ････> menu1.bas
110 ' 1985.2.10
120 ' ｵﾃﾅ ｲﾝｻﾂ ﾒﾆｭｰ ﾌﾟﾛｸﾞﾗﾑ
130 ' save "menu1.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 '
180 READ MENU
190 DIM MENU$(MENU),PROG$(MENU)
200 FOR I=1 TO MENU
210   READ MENU$(I),PROG$(I)
220 NEXT I
230 '
240 PRINT TAB(18)
250 PRINT STRING$(40,149)
260 PRINT TAB(28)
270 PRINT" ｱﾃﾅ ｲﾝｻﾂ ﾒﾆｭｰ"
280 PRINT TAB(18)
290 PRINT STRING$(40,149)
300 PRINT
310 '
320 FOR I=1 TO MENU
330   LOCATE 25,I+3 : PRINT MENU$(I)
340 NEXT I
350 '
360 LOCATE 18,12
370 PRINT STRING$(40,149)
380 LOCATE 23,13
390 PRINT "ｼｺﾞﾄ ﾊﾞﾝｺﾞｳ ｦ ﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ"
400 N$=INKEY$ : IF N$="" THEN GOTO 400
410 N=VAL(N$)
420 IF N>MENU OR N=0 THEN BEEP : GOTO 400
430 '
```

```
440 COLOR 4
450 LOCATE 25,N+3
460 PRINT MENU$(N)
470 COLOR 0
480 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
490 LOCATE 20,13 : PRINT SPC(40)
500 COLOR 6 : LOCATE 25,13
510 PRINT " ｼﾞｯｺｳ [,]  ｷｬﾝｾﾙ [.]  "
520 COLOR 0
530 I$=INKEY$ : IF I$="" THEN GOTO 530
540 IF I$="," AND N=MENU THEN CLS : END
550 IF I$="," THEN RUN PROG$(N)
560 IF I$<>"." THEN BEEP : GOTO 530
570 LOCATE 25,N+3 : PRINT MENU$(N)
580 LOCATE 20,13 : PRINT SPC(40)
590 GOTO 380
600 '
610 '―――――――――――――――――――――――――――――
620 '  menu data
630 '―――――――――――――――――――――――――――――
640 DATA 5
650 DATA ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ ･･････････1,dmail1.bas
660 DATA ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ ･････････2,dmail2.bas
670 DATA ｱﾃﾅ ﾉ ｲﾝｻﾂ ･･････････3,dmail3.bas
680 DATA ﾏｽﾀｰﾒﾆｭｰ ﾍ ﾓﾄﾞﾙ ･･････4,MSMENU
690 DATA ｵﾜﾘ ････････････････････5,dummy
```

入力できたら、まず**セーブ**です。

```
save "menu1.bas" ⏎
```

プログラムのセーブが終ったら、メニュープログラムを実行して、いろいろな処理を行ってみてください。

第2章 もう少しきめ細かに

2·1　ファイルに、より資料性を

第１章のテーマ、「セクタのもう１つの使い方」いかがでしたか？実務面でデータをより有効に活用しようという点ではまだまだという感じがしますが、前編、

『PC-8801MKⅡFR／MRデータファイル入門１』

を読んでいただいた方々に、もう一歩進めてセクタの使い方を理解していただくことが目的でしたので、ある程度目的は達成できたのではないかと考えています。

また、プログラミングには前の本から引き続いて**宛て名書き**を選びましたが、ディスクの扱い方をマスターするためには、できたら１つのテーマを通して、いろいろなケースを考察したほうがよいと考えたからです。というのは、そのほうが、フィールド設定でも、プログラムの流れでも、あるいはランダムファイルのフォーマットの作り方でも、くり返し学ぶことになりますので、それだけ体得がはやいと思います。

よく英語などをマスターするのでも、いろいろな本を読むよりは、１、２冊の本を暗記するくらいに読めといわれます。ディスクの勉強でも原理はまったく同じだと思います。

そんなわけで、本章でもやはり宛て名書きのテーマにそくして、今度はもっと実用性をもたせるようにしたいと思います。

データファイルのフォーマットを決める

前章では、男女別、あるいは特定の人の宛て名を表示しましたが、この節ではもう少し範囲を広げて、職業別あるいは年齢制限などを加えてみます。

プログラムの流れ自体はそれほど変わりませんが、ランダムファイルのフォーマットが異なってきますので、前節のプログラムの一部を流用せず、作りなおしたほうがはやいようです。

ランダムファイルで業務管理をする場合、プログラミングの前にまず、どのような処理をしたいかを決め、それにもとづいてランダムファイルの形をしっかり作っておかなければなりません。

この節ではいったいどこまで達成できるかわかりませんが、**処理してみたい項目**を列挙し、私たちの当面の目的としておきましょう。

フォーマット
●フォーマットとは形、様式などのことで、ディスクの場合、セクタ内をどのように使うかという割り当てといったところです。

❶　**管理データを500人分にふやしたい。**

❷　**データ入力をし、修正もしたい。**

❸　**男女別、職業別、年齢別に宛て名を表示したい。**

❹　**たとえば毎年４月１日をベースにして、登録データの年齢を更改したい。**

あげればまだまだいろいろあると思いますが、だいたいこんなところを目ざして進みたいと思います。

この節で扱うランダムファイルディスクリプタは、

2 : mail02. dat

とし、またインデックス用のシーケンシャルファイルのファイルディスクリプタは、

2 : indx02. dat

とします。前節のファイル名とは数字が１文字違うだけですから、間違わないように注意してください。

さて、ランダムファイルの設計ですが、次ページに概略を図示します。コメントを入れてありますので理解できると思いますが、簡単に説明しておきましょう。

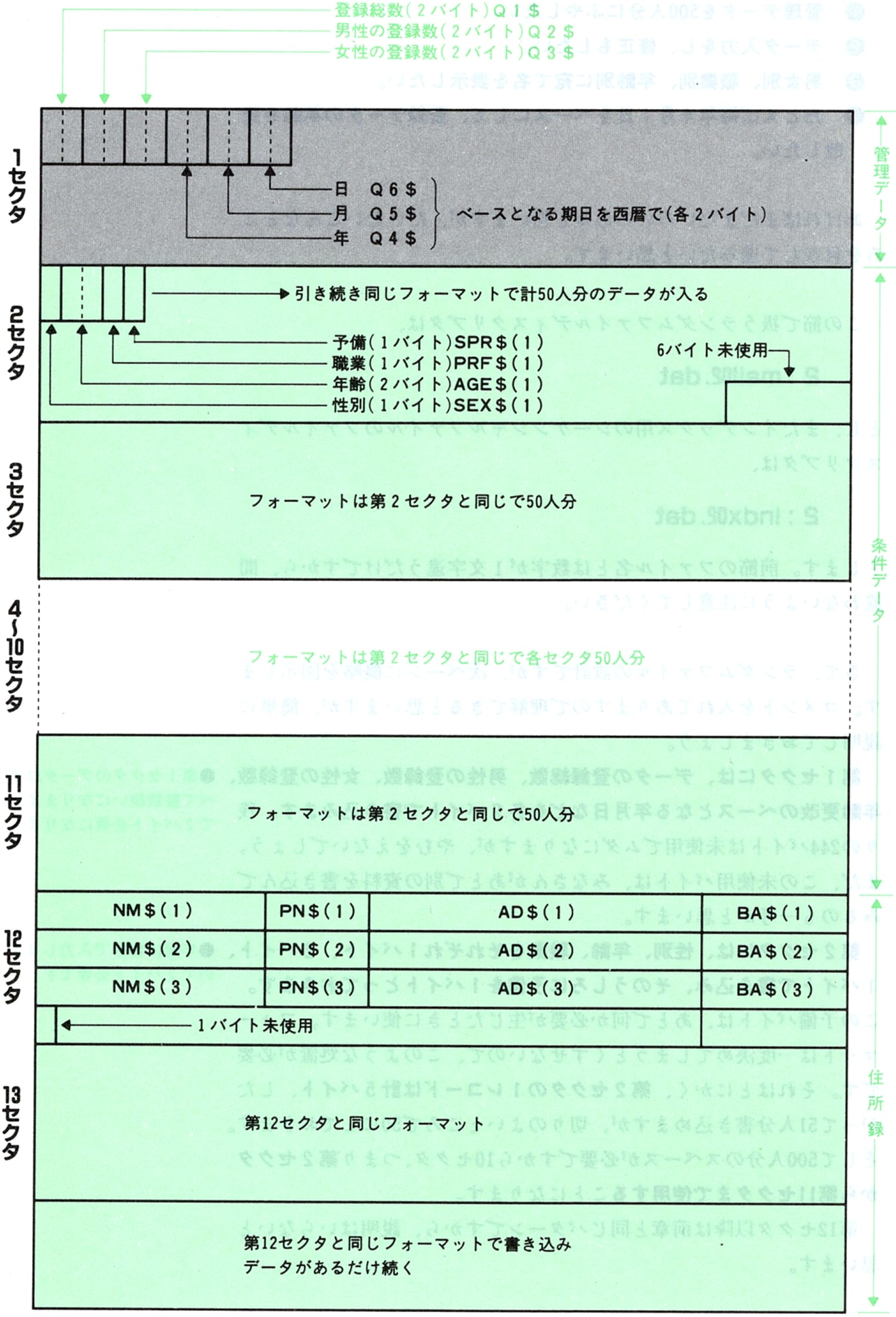
第１セクタには、データの登録総数、男性の登録数、女性の登録数、年齢更改のベースとなる年月日などを各２バイトで書き込みます。残りの244バイトは未使用でムダになりますが、やむをえないでしょう。まだ、この未使用バイトは、みなさんがあとで別の資料を書き込んでみるのも一考かと思います。

●第１セクタのデータはすべて整数扱いになりますので２バイト必要になります。

第２セクタには、性別、年齢、職業をそれぞれ１バイト、２バイト、１バイトで書き込み、そのうしろに予備を１バイトとっておきます。この予備バイトは、あとで何か必要が生じたときに使います。フォーマットは一度決めてしまうとくずせないので、このような処置が必要です。それはとにかく、**第２セクタの１レコードは計５バイト**、したがって51人分書き込めますが、切りのよいところで50にしておきます。そして500人分のスペースが必要ですから10セクタ、つまり**第２セクタから第11セクタまで使用する**ことになります。

●年齢は整数で入力しますので２バイト必要です。

第12セクタ以降は前章と同じパターンですから、説明はいらないと思います。


登録総数（2バイト）Q1$
男性の登録数（2バイト）Q2$
女性の登録数（2バイト）Q3$
1セクタ
日　Q6$
月　Q5$
年　Q4$
ベースとなる期日を西暦で（各2バイト）
管理データ
2セクタ
引き続き同じフォーマットで計50人分のデータが入る
予備（1バイト）SPR$(1)
職業（1バイト）PRF$(1)
年齢（2バイト）AGE$(1)
性別（1バイト）SEX$(1)
6バイト未使用
3セクタ
フォーマットは第2セクタと同じで50人分
4〜10セクタ
フォーマットは第2セクタと同じで各セクタ50人分
11セクタ
フォーマットは第2セクタと同じで50人分
条件データ
12セクタ
NM$(1)
PN$(1)
AD$(1)
BA$(1)
NM$(2)
PN$(2)
AD$(2)
BA$(2)
NM$(3)
PN$(3)
AD$(3)
BA$(3)
1バイト未使用
13セクタ
第12セクタと同じフォーマット
第12セクタと同じフォーマットで書き込み
データがあるだけ続く
住所録

2・2 データ書き込みプログラム

それではこれからプログラムをキーインしていきますが、先ほども申し上げましたように、プログラムの流れは前章とだいたい同じですので、詳しい説明は省略します。

わからないところがあったら、もう一度前章を参照してください。

セクタ番号、レコード番号を決める

最初に、**あるデータの、登録してある先頭データから数えて何番目かがわかったとき、これをもとにGETすべきセクタ番号およびそのセクタ内でのレコード番号を算出する式**を考えてみましょう。

ここでは仮にX番目のデータとしておきます。

この方法については28〜30ページでくわしく説明しましたので、もしわからなかったら、その箇所を読みなおしてください。

まず、**X番目のデータが書き込まれるべき第2〜第11セクタ内でのセクタ番号**は、次の式によって求められます。

セクタ番号1＝(X－1) ¥ 50＋2

セクタ番号の計算

1セクタ内に50レコード書き込まれますから、Xから1を引いた数を50で割ると、その商はスタートラインとなるセクタ（この場合第2セクタ）から数えて何番目かを表わしていますから（スタートラインのセクタを0番目とする）、これにスタートラインのセクタ番号を加えておけばよいですね。

では、**該当セクタ内でのレコード番号**はどうなるでしょうか。

Xから1を引いた数を50で割るとその端数（余り）は、1つ前にずれていますので、これに1を加えておけばよいですね。したがって、次の算式で計算できます。

レコード番号1＝(X－1) MOD 50＋1

レコード番号の計算

●この場合、性別、年齢、職業、予備を1レコードとして1セクタに50人分書き込みますので、ここでいうレコード番号とは1つのセクタ内で何番目に位置するかということです。

次に、**同じX番目のデータの第12セクタ以降での書き込みセクタ番号と、そのセクタ内でのレコード番号**はどうなるでしょうか。

いまと同じように考えてみてください。

今度は第12セクタが基準（スタートライン）になりますから、セクタ番号は次のようになります。

セクタ番号2＝(X－1) ¥ 3＋12

同様に、そのセクタ内でのレコード番号は、

レコード番号2＝(X－1) MOD 3＋1

となります。

DEF FN

これで初期設定の下準備ができました。前章で学んだDEF FN文を使って、セクタ番号、レコード番号を計算する関数を定義しておきましょう。

初期設定

この部分のプログラムを次に示します。

```
100 ' file name ････> DMAIL1.BAS
110 ' 1985.2.15
120 ' ランダ゛ム ファイル ヘ ノ カキコミ
130 ' save "DMAIL1.BAS"
140 '
150 '────────────────────────────
160 *SHOKISETTEI
170 '────────────────────────────
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DEF FNSEC1(X)=(X-1) ¥ 50+2      ┐
210 DEF FNREC1(X)=(X-1) MOD 50+1    ├─関数の定義
220 DEF FNSEC2(X)=(X-1) ¥ 3+12      │
230 DEF FNREC2(X)=(X-1) MOD 3+1     ┘
240 DIM NM$(3),PN$(3),AD$(3),BA$(3)
250 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
260 '
```

〈注〉250行の変数名のうちPRFはPROFESSIONの略、SPRはSPAREの略です。別に何を使ってもかまいませんが関連性をもたせたほうがわかりやすいと思います。

今度は**画面の行数を25行**にしました。WIDTH文の第2パラメータ、CONSOLE文の第2パラメータを25にしてください。

また、プログラムファイル名は前回と違えて大文字にしました。

年齢、職業、予備のための配列変数をあらたに宣言しますが、1セクタに50人までの書き込みですから大きさは50でいいですね。

なお、160行にラベル名を使っていますが、これはプログラム全体を見やすくするためのもので、別に意味はありません。

ファイルのオープンとフィールド定義

この部分も、もうおわかりと思いますので、プログラムを示し、必要に応じて説明しておきます。

❶ 第2セクタから第11セクタまでのためのFIELD定義

```
2000 '─────────────────────
2010 *FOPEN1
2020 '─────────────────────
2030 OPEN "2:mail02.dat" AS #1
2040 FOR I=1 TO 50
2050   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
         2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
2060 NEXT I
2070 RETURN
2080 '
```

ランダムファイルを1番バッファを使ってOPENし、**性別、年齢、職業、予備のデータやりとりに使用する配列変数のフィールド幅を、ダミー変数を使って多重定義して設定**しています。

50人分ですから、50回ループさせればよいですね。そして1回ループするたびに**ダミー変数のフィールド幅**は5きざみにふくれあがり、**最終的には245バイトに設定**されます。

前にも述べましたが、GETしてこのダミー変数を使わなければ何も問題は起こりません。

❷ 第12セクタ以降のためのFIELD定義

```
2500 '─────────────────────
2510 *FOPEN2
2520 '─────────────────────
2530 OPEN "2:mail02.dat" AS #2
2540 FOR I=1 TO 3
2550   FIELD #2,85*(I-1) AS D$,20 AS NM$(I),
            6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
2560 NEXT I
2570 RETURN
2580 '
```

このサブルーチンは前章とまったく同じです。

２番のバッファを使います。

OPEN文のファイル番号と、FIELD文のバッファ番号を同じにしておいてください。当然のことなのですが、よくこれを間違えたりして、**正しいデータが得られないケースが多い**ようです。

❸ 第１セクタのデータやりとりのためのFIELD定義

```
3000 '————————————————————————————————
3010 *FOPEN3
3020 '————————————————————————————————
3030 OPEN "2:mail02.dat" AS #1
3040 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
            2 AS Q4$,2 AS Q5$,2 AS Q6$
3050 RETURN
3060 '
```

106ページの図を見ていただけばわかると思いますが、登録データ数、男性の登録数、女性の登録数、基準となる年、月、日用の変数に対して、それぞれ２バイト幅でフィールドを割り当てています。

使用するバッファは１番ですから、❶の場合と重複しますが、同時にファイルをOPENしないようにしますので支障はありません。

データ入力までの下準備

ランダムファイルオープンとフィールド設定のサブルーチンができましたので、次にデータ入力までの下準備をしておきます。

この部分でやるべきことは、いま作ったサブルーチンにジャンプさせてファイルをOPENするのはいうまでもありませんが、そのほかに、**初回のデータ入力でなければ第１セクタに書き込まれている男性および女性の登録数をとりだしておかなければなりません**（これはデータ入力のたびに加算し、終ったら第１セクタにふたたび書き込みます）。

また、初回の書き込みの場合には、インデックス用のシーケンシャルファイルを**OUTPUTモードでOPEN**、２回目以降なら**APPENDモードでOPEN**する必要がありますね。

そして最後に、ベースとなる年月日の入力です。

さあ、それではプログラムを次ページに示しますので、さっそくキーインしてみましょう。

```
270 '─────────────────────────────
280 *JYUNBI
290 '─────────────────────────────
300 GOSUB *FOPEN3
310 IF LOF(1)=0 THEN CLOSE #1 : GOTO 360
320 GET #1,1
330 OTOKO=CVI(Q2$)
340 ONNA =CVI(Q3$)
350 CLOSE #1
360 GOSUB *FOPEN1
370 GOSUB *FOPEN2
380 IF LOF(1)=0 THEN PUT #1,11 : GOTO 420
390 GOSUB *YOMU : P=COUNT
400 OPEN "2:indx02.dat" FOR APPEND AS #3
410 GOTO 430
420 OPEN "2:indx02.dat" FOR OUTPUT AS #3
430 '
440 LOCATE 30,0 : COLOR 4
450 PRINT "     ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ     " : COLOR 0
460 LOCATE 20,1 : PRINT STRING$(40,149)
470 LOCATE 20,18: PRINT STRING$(40,149)
480 '
490 GOSUB *HIZUKE
500 '
```

初回のデータ入力でなければ第1セクタをGET（320行）

男女の登録数をカウンタに代入（330〜340行）

第2〜第11セクタをGETできるようにあらかじめ第11セクタをPUT（380行）

ジョブ名表示（440〜450行）

画面の上下に罫線を引く（460〜470行）

日付の入力（490行）

ファイル長のチェック

300行でランダムファイルをOPENし、310行で**ファイル長をチェックし、もし0ならば初めての書き込み**ですからCLOSEしてからジャンプさせます。

初回の書き込みではなければ、すでに男女の登録数が第1セクタにセットされているわけですから、これを取り出して**カウンタ用変数に格納**しておきます。320行から350行までがこの処理です。

360行、370行では、ランダムファイルをOPENし、第2セクタ以降および第12セクタ以降にデータを書き込むためにフィールド定義をサブルーチンで行ないます。

ランダムファイルをGETするためには、あらかじめPUTしてあることが必要条件です。しかしこれは、GETするセクタをあらかじめPUTするという意味では必ずしもなく、**GETしたいセクタよりもっと先のセクタをPUTすれば、最初のセクタからこのセクタまでは常にGETすることができます。**

●この本の校正時に気づいたのですが、380行のPUT文では第11セクタをPUTしなくても第2セクタをPUTしておけば十分なようです。というのは、2回目以降の書き込みの場合、少なくとも第12セクタはPUTされているので、第3セクタ以降はいつでもGETできるからです。

つまり、初回のデータ入力時に、一番終りの11セクタをPUTしてし

まえば、第２～第11セクタはいつでもGETできるのです。

390、400行は、２回目以降のデータ入力の場合のデータカウント処理とシーケンシャルファイルのOPENです。

490行のサブルーチンは、ベースとなる年月日を入力します。**この日付は、毎年一度、年齢の更新をしたいので、ある年度内では常に期首の年月日をキーインする**ようにします。たとえば、1985年４月１日を期首にしましたら、1986年３月31日までは1985年４月１日をベースとします。

シーケンシャルファイルの読み飛ばし

次は、２回目以降の書き込みの場合、すでに登録ずみのデータ数をカウントしなければなりませんので、次の読み飛ばしサブルーチンを入れておきます。

```
3500 '─────────────────────────
3510 *YOMU
3520 '─────────────────────────
3530 OPEN "2:indx02.dat" FOR INPUT AS #3
3540 WHILE NOT EOF(3)
3550   COUNT=COUNT+1
3560   INPUT #3,DUMMY
3570 WEND
3580 CLOSE #3
3590 RETURN
3600 '
```

日付の入力

これもサブルーチンにしてしまいます。プログラムリストを右ページに示します。

これは１年ごとに年齢の更新をしたいので入力するようにしました。入力は西暦何年何月です。4080行は誤って昭和何年で入力してしまった場合の処理です。

```
4000 '────────────────────────────
4010 *HIZUKE
4020 '────────────────────────────
4030 GOSUB *BUZZER
4040 LOCATE 27,19 : COLOR 6
4050 PRINT " ﾍﾞｰｽ ﾉ 年月日 ｦ ﾄﾞｳｿﾞ(ｾｲﾚｷ) "
4060 COLOR 0
4070 LOCATE 25,3 : INPUT "ﾅﾝ年";YY$
4080 IF LEN(YY$)<4 THEN BEEP : GOTO 4070
4090 LOCATE 25,5 : INPUT "ﾅﾝ月";MM$
4100 LOCATE 25,7 : INPUT "ﾅﾝ日";DD$
4110 GOSUB *BUZZER
4120 GOSUB *KESU1 : GOSUB *KESU2
4130 LOCATE 25,3 : PRINT "ﾍﾞｰｽ ﾉ 年月日 :";
4140 PRINT YY$"年 "MM$"月 "DD$"日"
4150 LOCATE 27,19 : COLOR 6
4160 PRINT "  ﾏﾁｶﾞｲ ﾅｲﾃﾞｽｶ ( y/n )?  "
4170 COLOR 0
4180 I$=INKEY$ : IF I$="" THEN GOTO 4180
4190 IF I$="Y" OR I$="y" THEN GOTO 4220
4200 IF I$="N" OR I$="n" THEN GOTO 4240
4210 BEEP : GOTO 4180
4220 YY=VAL(YY$) : MM=VAL(MM$) : DD=VAL(DD$)
4230 GOSUB *KESU1 : GOSUB *KESU2 : RETURN
4240 GOSUB *KESU1 : GOSUB *KESU2 : GOTO 4040
4250 '
```

(4050：画面の下へ反転表示／4080：西暦でなかった／4110：ブザー音／4130〜4160：確認／4220：数値に変換)

このサブルーチンの中で、さらに3つのサブルーチンのラベル名が見られます。

❶ ＊**BUZZER**……ブザー音を鳴らします
❷ ＊**KESU1**………画面の一部を消去します
❸ ＊**KESU2**………オペレーションメッセージ行を消去します

❶は**ブザー音**（ビープ音）を鳴らすサブルーチンです。よく使うのでサブルーチン扱いにしました。

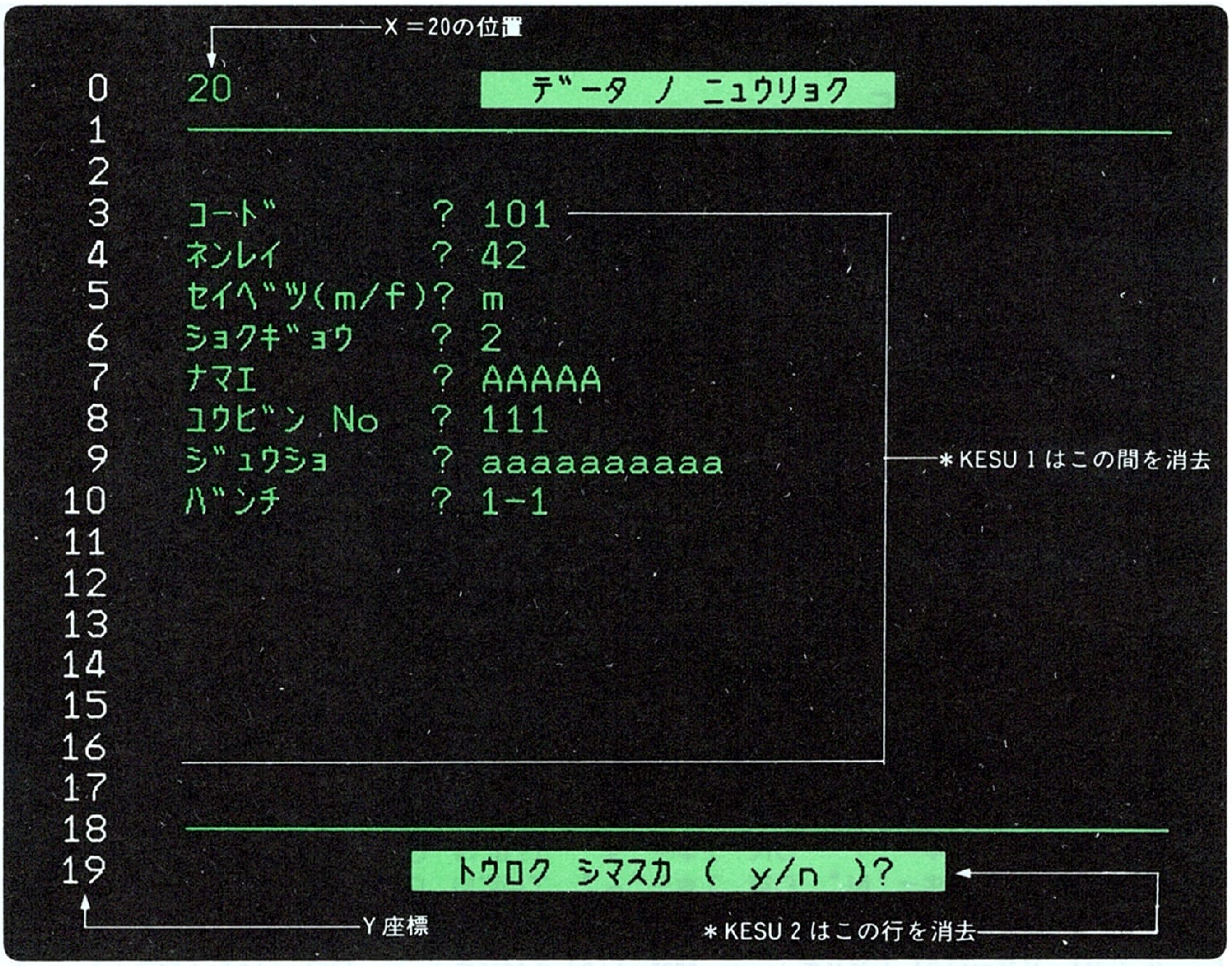
❷および❸は**画面の一部を消去するサブルーチン**です。これもよく使うのでサブルーチンにしました。

次ページにプログラムを実行中の図を示しますので参照してください。様子がよくわかると思います。一番左端の縦列の数字はわかりやすいように入れたものです。プログラムとは関係ありません。

ブザー音

画面の部分消去

●CLSを使うと画面全部が消されてしまいますのでもう一度必要部分を出力しなければなりません。一部消去の方法も覚えておくと便利です。



それでは、いま説明したサブルーチンをキーインしてしまいましょう。

最初は画面を消去するサブルーチンからです。

```
4500 '―――――――――――――――――――――――――――
4510 *KESU1
4520 '―――――――――――――――――――――――――――
4530 FOR I=3 TO 16
4540   LOCATE 20,I : PRINT SPC(40)
4550 NEXT I
4560 RETURN
4570 '
5000 '―――――――――――――――――――――――――――
5010 *KESU2
5020 '―――――――――――――――――――――――――――
5030 LOCATE 20,19 : PRINT SPC(40)
5040 RETURN
5050 '
```

次は**ブザー音のサブルーチン**です。

```
5500 '―――――――――――――――――――――――――――
5510 *BUZZER
5520 '―――――――――――――――――――――――――――
5530 FOR I=1 TO 300 : BEEP 1 : BEEP 0 : NEXT I
5540 RETURN
5550 '
```

データ入力部分

これで一応前準備が終りましたので、いよいよメインのデータ入力ルーチンです。プログラムを次に示します。

```
510 '―――――――――――――――――――――――――――
520 *LOOP
530 '―――――――――――――――――――――――――――
540 LOCATE 40,3 : COLOR 6
550 PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  " : COLOR 0
560 LOCATE 20,3 : INPUT "ｺｰﾄﾞ         ";CD  ←――コード入力
570 IF CD=9999 THEN GOTO *OWARI
580 LOCATE 40,3 : PRINT SPC(30)
590 LOCATE 20,4 : INPUT "ﾈﾝﾚｲ         ";A0
600 LOCATE 20,5 : INPUT "ｾｲﾍﾞﾂ(m/f)";A1$
610 GOSUB *SHOKUGYO ←――――――――職業入力サブルーチンへ
620 LOCATE 20,7 : INPUT "ﾅﾏｴ         ";A2$
630 LOCATE 20,8 : INPUT "ﾕｳﾋﾞﾝ No    ";A3$
640 LOCATE 20,9 : INPUT "ｼﾞｭｳｼｮ      ";A4$
650 LOCATE 20,10: INPUT "ﾊﾞﾝﾁ        ";A5$
660 GOSUB *BUZZER ←―――――――――――ブザー音
670 LOCATE 30,19 : COLOR 4
680 PRINT " ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?  " :COLOR 0
690 I$=INKEY$ : IF I$="" THEN GOTO 690
700 IF I$="Y" OR I$="y" THEN GOTO *KAKU ←――書き込み
710 IF I$="N" OR I$="n" THEN GOTO *TORIKESHI ←┐
720 BEEP : BEEP : GOTO 690                     │
730 '                              取り消し ――――┘
```

コードに「9999」が入力されると終りの処理へジャンプします。それ以外の場合は年齢、性別と、順番に入力していきます。

610行のサブルーチンへのジャンプは、職業を入力させるものです

が、これは、職業メニューを表示して、その中から番号選択によって行わせます。サブルーチンはあとで示します。

番地までのデータ入力が終ると、登録するかどうかを聞いてきます。登録する場合には書き込みルーチンへ、そうでない時は取り消しのルーチンへジャンプします。

職業メニューサブルーチン

610行からのジャンプ先のサブルーチンを次に示します。

```
6000 ´────────────────────────────────
6010 *SHOKUGYO
6020 ´────────────────────────────────
6030 GOSUB *BUZZER
6040 GOSUB *KESU1                              番号入力待ち
6050 LOCATE 30,19: COLOR 4
6060 PRINT " ｼｮｸｷﾞｮｳ ﾊﾞﾝｺﾞｳｦ ﾄﾞｳｿﾞ " : COLOR 0
6070 LOCATE 30,3 : PRINT "ｶｲｼｬ ﾔｸｲﾝ ··········1"
6080 LOCATE 30,5 : PRINT "ｻﾗﾘｰﾏﾝ ············2"
6090 LOCATE 30,7 : PRINT "ｶﾞｸｾｲ ·············3"
6100 LOCATE 30,9 : PRINT "ｼｭﾌ ···············4"
6110 LOCATE 30,11: PRINT "ｿﾉﾀ ···············5"
6120 PF$=INKEY$ : IF PF$="" THEN GOTO 6120
6130 GOSUB *KESU1 : GOSUB *KESU2  ←画面の一部を消去
6140 LOCATE 20,3 : PRINT "ｺｰﾄﾞ       ?";CD
6150 LOCATE 20,4 : PRINT "ﾈﾝﾚｲ       ?";A0
6160 LOCATE 20,5 : PRINT "ｾｲﾍﾞﾂ(m/f)? ";A1$
6170 LOCATE 20,6 : PRINT "ｼｮｸｷﾞｮｳ    ? ";PF$
6180 RETURN                                    データを再表示
6190 ´
```

ごらんのように、1から5までの職業を表示し、番号を選択させます。

職業については、この本が終ってから、みなさんにとって役立つように分類してみてください。

6120行のINKEY＄で番号入力を受け取ると、**画面を一部クリアして、前ページのリスト620行の名前入力につながるように、コード、年齢、性別、職業を表示してからリターン**させます。

データの書き込み部分

１人分のデータ入力が終りましたら、次は**ランダムファイルへのデータ書き込み**です。

プログラムは次のとおりです。

```
740 '―――――――――――――――――――――――――――――――――
750 *KAKU
760 '―――――――――――――――――――――――――――――――――
770 IF A1$="m" OR A1$="M" THEN OTOKO=OTOKO+1
780 IF A1$="f" OR A1$="F" THEN ONNA=ONNA+1
790 P=P+1
800 SEC1=FNSEC1(P)
810 REC1=FNREC1(P)
820 SEC2=FNSEC2(P)
830 REC2=FNREC2(P)
840 IF REC2>1 THEN GET #2,SEC2
850 GET #1,SEC1
860 WRITE #3,CD
870 IF REC2<3 THEN GOSUB *BUFCLEAR
880 LSET SEX$(REC1)=A1$
890 LSET AGE$(REC1)=MKI$(A0)
900 LSET PRF$(REC1)=PF$
910 LSET NM$(REC2) =A2$
920 LSET PN$(REC2) =A3$
930 LSET AD$(REC2) =A4$
940 LSET BA$(REC2) =A5$
950 PUT #1,SEC1 : PUT #2,SEC2
960 GOSUB *KESU1 : GOSUB *KESU2 : GOTO *LOOP
970 '
```

800〜830行：書き込みセクタとそのセクタ内でのレコード番号を計算

840行：第12セクタ以降への書き込みでレコード番号が２または３の場合そのセクタをGETしておく

860行：コードの書き込み

880〜940行：バッファ上でデータセット

950行：ファイルへ書き込む

960行：画面を一部クリアしてからループ

770行、780行は男性あるいは女性の場合、それぞれのカウンタを１つずつふやしています。

800行から830行で、初期設定で定義した関数を使って、書き込みセクタとそのセクタ内でのレコード番号を算出しています。

セクタ内でのレコード番号が２または３の場合、840行でやっているように、そのセクタをGETしておきませんと以前のデータが消されてしまう心配があります。

バッファクリア

870行は**バッファクリア**です。これは、セクタ内でのレコード番号が１または２の場合に行います。48〜50ページでこの点については詳しく説明してありますので、そちらを参照ください。

880行から900行までは**1番バッファでLSET**、910行から940行までは**2番バッファでLSET**しています。そして、このセットされたデータを920行でPUTしているわけです。

バッファクリア

次に**バッファクリアのサブルーチン**を示しますので入力してください。

```
6500 '―――――――――――――――――――――――
6510 *BUFCLEAR
6520 '―――――――――――――――――――――――
6530 FOR I=REC2+1 TO 3
6540   LSET NM$(I)=""
6550   LSET PN$(I)=""
6560   LSET AD$(I)=""
6570   LSET BA$(I)=""
6580 NEXT I
6590 RETURN
```

ここで注意したいことは、4つの配列変数

NM$, PN$, AD$, BA$

ヌルストリング

をクリアするためには、それぞれに**ヌルストリングを代入する**わけですが、これらがバッファ上で効力を持つ変数なので、必ずLSET文によって行ってください。つまり、単に、

```
100 FOR I=REC2+1 TO 3
110     NM$(I)=""
120     PN$(I)=""
130     AD$(I)=""
140     BA$(I)=""
150 NEXT I
```

としたのでは意味を持ちません。

入力データのキャンセル

次は、115ページの行番号710のジャンプ先のサブルーチンです。プログラムリストは次ページに示します。

この場合には、画面を一部消去して、コード入力からやりなおしてす。

```
980 '
990 *TORIKESHI
1000 '
1010 GOSUB *KESU1 : GOSUB *KESU2
1020 GOTO *LOOP
1030 '
```

終りの処理

コードに「9999」が入力されると、このルーチンへジャンプしてきます。

```
1040 '
1050 *OWARI
1060 '
1070 CLOSE
1080 GOSUB *FOPEN3          ←ファイルオープン
1090 LSET Q1$=MKI$(P)
1100 LSET Q2$=MKI$(OTOKO)   ┐
1110 LSET Q3$=MKI$(ONNA)    │
1120 LSET Q4$=MKI$(YY)      ├必要データをセット
1130 LSET Q5$=MKI$(MM)      │
1140 LSET Q6$=MKI$(DD)      ┘
1150 PUT #1,1 : CLOSE #1    ←書き込み
1160 CLS
1170 RUN "menu2.bas"
1180 '
```

第1セクタに、登録数、男性、女性の数、年月日などを書き込んでから、メニュープログラムを実行します。

さあ、それではプログラムをセーブしてから終りにしましょう。

```
save "DMAIL1.BAS" ⏎
```

2·3　プログラムの実行

プログラムをRUNさせて実行する前に、画面がどのように変わっていくか簡単に説明しておきましょう。

次の順序で処理が流れます。

処理の流れ

❶　最初に図1のような画面になり、ベースとなる年月日の入力をします。年は西暦でキーインしてください。

❷　ベースとなる年月日の入力が終ると、いったん確認のメッセージが表示されます。ノーならやりなおしです。

❸　間違いなければ、図2のようにコード入力画面となります。終りたいときは「9999」を入力してください。

❹　年齢、性別まで入力すると次は職業の入力となり、図3のように自動的にメニューが表示されます。

該当する番号を入力してください。リターンキーを押す必要はありません。

❺　すると図4のような画面になり、名前から番地までを入力し、これでOKならディスクへ書き込まれ、これが1プロセスになります。

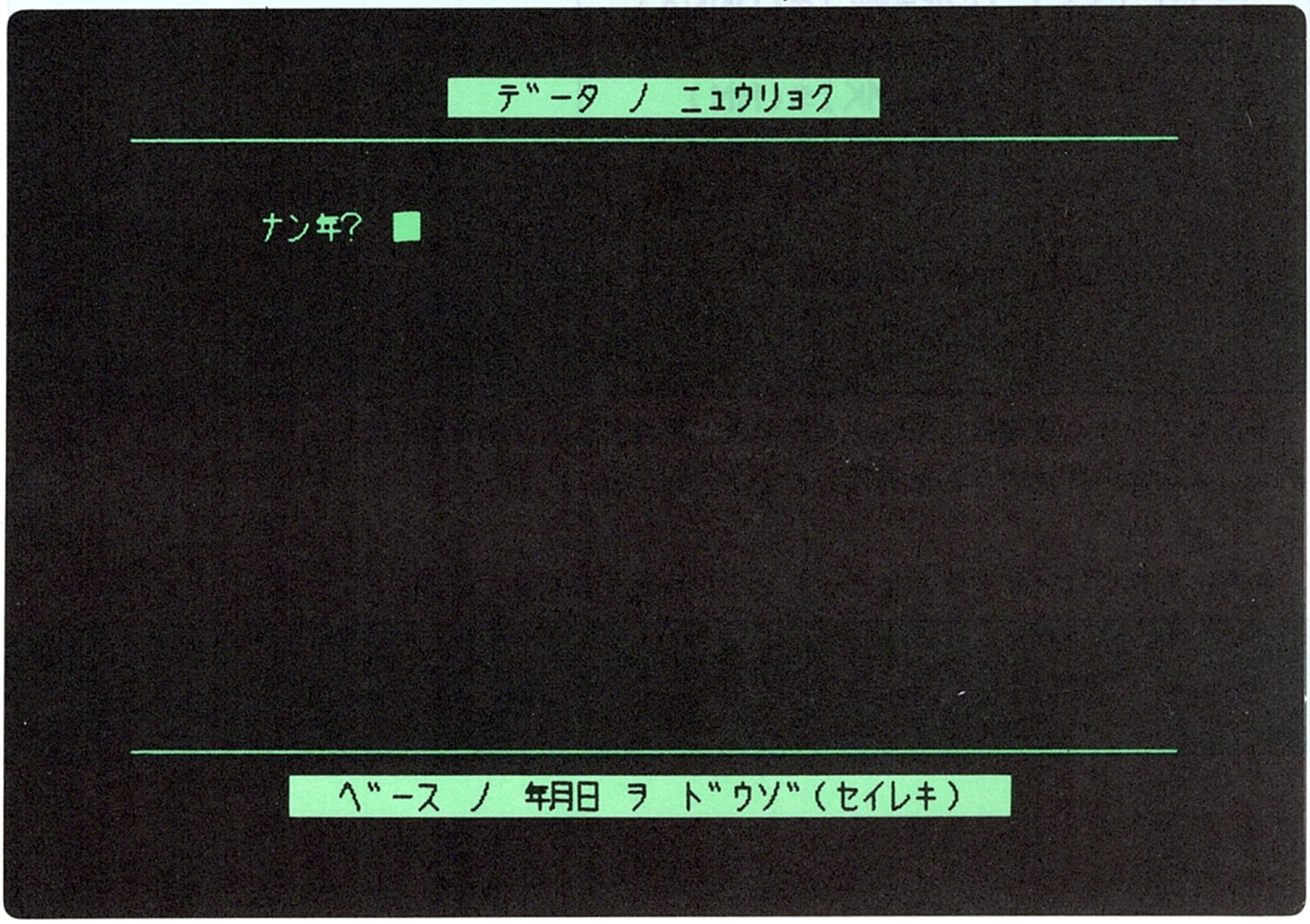


図1 ● ベースとなる年月日入力

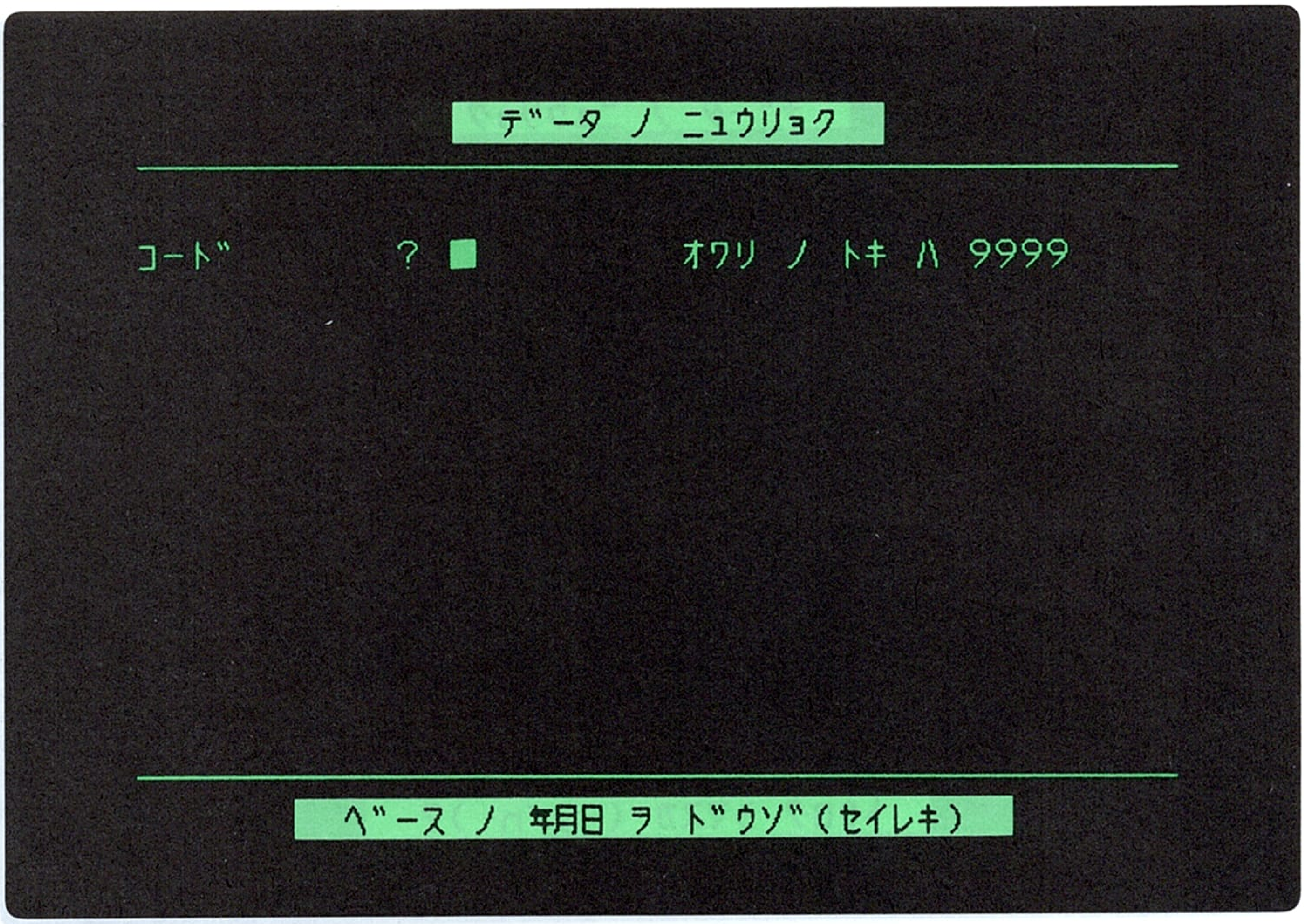


図2●コードの入力画面

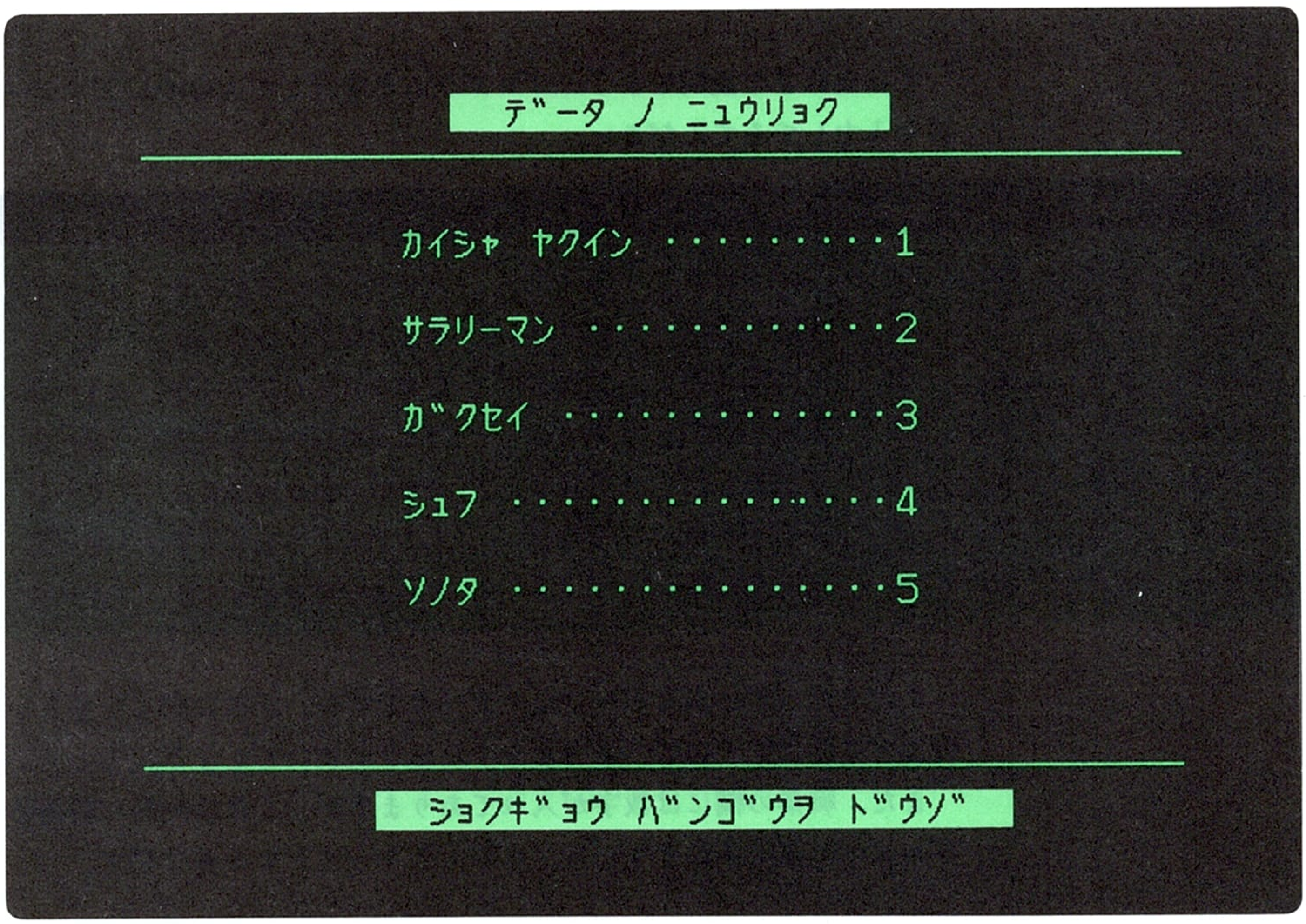


図3●職業メニュー画面

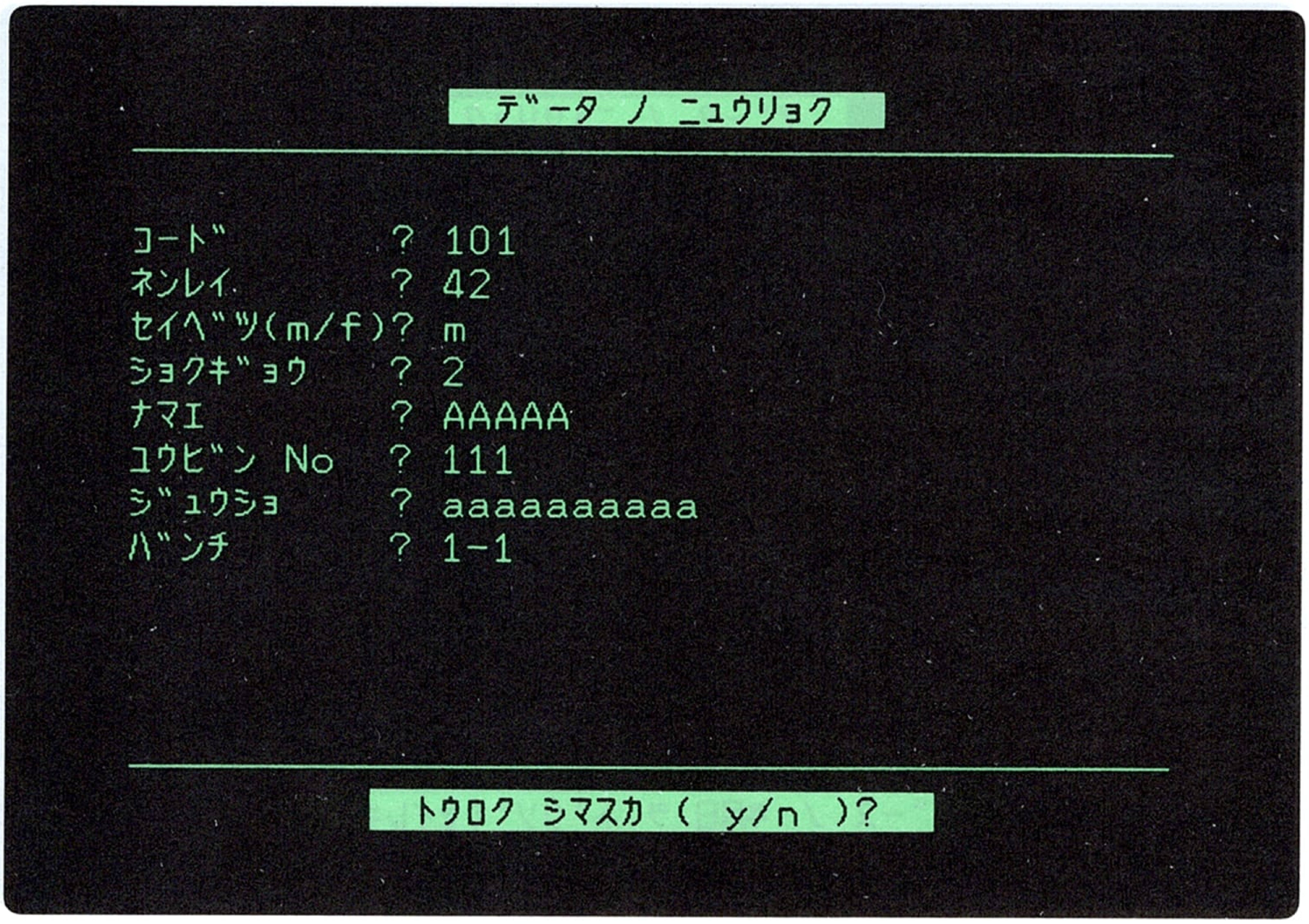


図4 ●登録するかどうかの確認画面

では先ほどの説明のようにプログラムを実行して次のサンプルデータを入力してみてください。

コード	年齢	性別	職業	氏　名	郵便番号	住　所	番地
101	42	m	2	AAAAA	111	aaaaaaaaaa	1−1
201	10	f	3	BBBBB	222	bbbbbbbbbb	2−1
102	50	m	1	CCCCC	333	cccccccccc	1−2
202	27	f	3	DDDDD	444	dddddddddd	2−2
103	30	m	4	EEEEE	555	eeeeeeeeee	1−3
203	19	f	3	FFFFF	666	ffffffffff	2−3
104	32	m	2	GGGGG	777	gggggggggg	1−4

またまた、ずいぶんそっけないデータですが、正常に動くかどうか確認するだけですので、入力しやすいほうがよいでしょう。

また、職業の欄には数字が入れてありますが、これは画面に職種のメニューが表示されますから、そのときキーインする番号です。

2·4 果たしてうまく書き込めたか?

プログラムがうまく動作したかどうか、つまり私たちが期待するようにディスクに書き込まれたかどうかテストしてみましょう。

ではプログラム

DMAIL1.BAS

の不要行を次のようにデリート((削除) します。

デリート (削除)

```
delete 100-230
Ok
delete 260-1180
Ok
delete 3050-6590
Ok
2070
2570
■
```

これはどういう意味だかわかりますか? 第1章を思いだしてください。

そうですね、ファイルオープンとフィールド定義のところだけ残しておくことです。そしてこれをRUNさせて、ダイレクトモードでデータ書き込みの確認をしょうというわけです。

RUNする前にもう2か所、3030行と3040行のバッファ番号を「3」に変更しておいてください。

```
3030 OPEN "2:mail02.dat" AS #3
3040 FIELD #3,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
```

なぜだかわかりますね。すでに2030行でバッファ1は使っているからです。なお、このファイルオープンのプログラムは次のようにセーブしておいてください。次節で使います。

```
save "DMTEST" ⏎
```

さあ、それでは実行してください。「Ok」が表示されたら、まず第2セクタの書き込み状況を調べてみましょう。

ダイレクトモードで次のようにキーインしてください。

```
run
Ok
get #1,2
Ok
for i=1 to 7:? sex$(i);cvi(age$(i));prf$(i):next
m 42 2
f 10 3
m 50 1
f 27 3
m 30 4
f 19 3
m 32 2
Ok
■
```

get #1,2 ← 第２セクタをGET

m 42 2 ～ m 32 2 ← 性別、年齢、職業の順に７人のデータが書き込まれている

どうですか？　間違いなく７人のデータが書き込まれていましたね。

では次に第13セクタをチェックしてみましょう。名前とか住所は12セクタから１セクタ３人あて書き込まれていますから、第13セクタというと、４人目から６人目までの人のデータということになります。

使用バッファは２番です。

次のようにキーインしてください。

```
get #2,13
Ok
for i=1 to 3:? nm$(i):? pn$(i):?ad$(i):? ba$(i):next
DDDDD
444
ddddddddd
2-2
EEEEE
555
eeeeeeeeee
1-3
FFFFF
666
fffffffff
2-3
Ok
■
```

DDDDD ～ 2-2 ← ４人目のデータ

EEEEE ～ 1-3 ← ５人目のデータ

FFFFF ～ 2-3 ← ６人目のデータ

これも異常なさそうです。

それでは第1セクタはどうでしょうか。

今度は3番バッファを使ってGETしてください。データは数値データでしたから、**CVIで変換して表示します。**

```
get #3,1
Ok
? cvi(q1$)cvi(q2$)cvi(q3$)cvi(q4$)cvi(q5$)cvi(q6$)
 7  4  3  1985  4  1
Ok
■
```

全データ数が7、うちわけは男性4、女性3、ベースの日付が1985年4月1日、これもうまくいっていました。

最後に必ずCLOSEをしておいてください。

なお、正しい書き込みデータが得られなかった方は、次にプログラムリストを示しますから、もう一度チェックし、ランダムファイルとシーケンシャルファイルをKILLコマンドで消してから、再度データ入力からやってみてください。

```
100 '  file name ････> DMAIL1.BAS
110 '  1985.2.15
120 '  ﾗﾝﾀﾞﾑ ﾌｧｲﾙ ﾍ ﾉ ｶｷｺﾐ
130 '  save "DMAIL1.BAS"
140 '
150 '----------------------------------------
160 *SHOKISETTEI
170 '----------------------------------------
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DEF FNSEC1(X)=(X-1) ¥ 50+2
210 DEF FNREC1(X)=(X-1) MOD 50+1
220 DEF FNSEC2(X)=(X-1) ¥ 3+12
230 DEF FNREC2(X)=(X-1) MOD 3+1
240 DIM NM$(3),PN$(3),AD$(3),BA$(3)
250 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
260 '
270 '----------------------------------------
280 *JYUNBI
290 '----------------------------------------
300 GOSUB *FOPEN3
310 IF LOF(1)=0 THEN CLOSE #1 : GOTO 360
320 GET #1,1
330 OTOKO=CVI(Q2$)
340 ONNA =CVI(Q3$)
350 CLOSE #1
360 GOSUB *FOPEN1
370 GOSUB *FOPEN2
380 IF LOF(1)=0 THEN PUT #1,11 : GOTO 420
390 GOSUB *YOMU : P=COUNT
400 OPEN "2:indx02.dat" FOR APPEND AS #3
410 GOTO 430
420 OPEN "2:indx02.dat" FOR OUTPUT AS #3
430 '
440 LOCATE 30,0 : COLOR 4
450 PRINT "    ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ   " : COLOR 0
460 LOCATE 20,1 : PRINT STRING$(40,149)
470 LOCATE 20,18: PRINT STRING$(40,149)
480 '
490 GOSUB *HIZUKE
500 '
510 '----------------------------------------
```

```
520 *LOOP
530 '──────────────────────────────
540 LOCATE 40,3 : COLOR 6
550 PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  " : COLOR 0
560 LOCATE 20,3 : INPUT "ｺｰﾄﾞ       ";CD
570 IF CD=9999 THEN GOTO *OWARI
580 LOCATE 40,3 : PRINT SPC(30)
590 LOCATE 20,4 : INPUT "ﾈﾝﾚｲ       ";A0
600 LOCATE 20,5 : INPUT "ｾｲﾍﾞﾂ(m/f)";A1$
610 GOSUB *SHOKUGYO
620 LOCATE 20,7 : INPUT "ﾅﾏｴ        ";A2$
630 LOCATE 20,8 : INPUT "ﾕｳﾋﾞﾝ No  ";A3$
640 LOCATE 20,9 : INPUT "ｼﾞｭｳｼｮ     ";A4$
650 LOCATE 20,10: INPUT "ﾊﾞﾝﾁ       ";A5$
660 GOSUB *BUZZER
670 LOCATE 30,19 : COLOR 4
680 PRINT " ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?  " :COLOR 0
690 I$=INKEY$ : IF I$="" THEN GOTO 690
700 IF I$="Y" OR I$="y" THEN GOTO *KAKU
710 IF I$="N" OR I$="n" THEN GOTO *TORIKESHI
720 BEEP : BEEP : GOTO 690
730 '
740 '──────────────────────────────
750 *KAKU
760 '──────────────────────────────
770 IF A1$="m" OR A1$="M" THEN OTOKO=OTOKO+1
780 IF A1$="f" OR A1$="F" THEN ONNA=ONNA+1
790 P=P+1
800 SEC1=FNSEC1(P)
810 REC1=FNREC1(P)
820 SEC2=FNSEC2(P)
830 REC2=FNREC2(P)
840 IF REC2>1 THEN GET #2,SEC2
850 GET #1,SEC1
860 WRITE #3,CD
870 IF REC2<3 THEN GOSUB *BUFCLEAR
880 LSET SEX$(REC1)=A1$
890 LSET AGE$(REC1)=MKI$(A0)
900 LSET PRF$(REC1)=PF$
910 LSET NM$(REC2) =A2$
920 LSET PN$(REC2) =A3$
930 LSET AD$(REC2) =A4$
```

```
940 LSET BA$(REC2) =A5$
950 PUT #1,SEC1 : PUT #2,SEC2
960 GOSUB *KESU1 : GOSUB *KESU2 : GOTO *LOOP
970 '
980 '―――――――――――――――――――――――――――――――――――――
990 *TORIKESHI
1000 '―――――――――――――――――――――――――――――――――――――
1010 GOSUB *KESU1 : GOSUB *KESU2
1020 GOTO *LOOP
1030 '
1040 '―――――――――――――――――――――――――――――――――――――
1050 *OWARI
1060 '―――――――――――――――――――――――――――――――――――――
1070 CLOSE
1080 GOSUB *FOPEN3
1090 LSET Q1$=MKI$(P)
1100 LSET Q2$=MKI$(OTOKO)
1110 LSET Q3$=MKI$(ONNA)
1120 LSET Q4$=MKI$(YY)
1130 LSET Q5$=MKI$(MM)
1140 LSET Q6$=MKI$(DD)
1150 PUT #1,1 : CLOSE #1
1160 CLS
1170 RUN "menu2.bas"
1180 '
2000 '―――――――――――――――――――――――――――――――――――――
2010 *FOPEN1
2020 '―――――――――――――――――――――――――――――――――――――
2030 OPEN "2:mail02.dat" AS #1
2040 FOR I=1 TO 50
2050   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
         2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
2060 NEXT I
2070 RETURN
2080 '
2500 '―――――――――――――――――――――――――――――――――――――
2510 *FOPEN2
2520 '―――――――――――――――――――――――――――――――――――――
2530 OPEN "2:mail02.dat" AS #2
2540 FOR I=1 TO 3
2550   FIELD #2,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
```

```
2560 NEXT I
2570 RETURN
2580 '
3000 '--------------------------------------------------------------
3010 *FOPEN3
3020 '--------------------------------------------------------------
3030 OPEN "2:mail02.dat" AS #1
3040 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
3050 RETURN
3060 '
3500 '--------------------------------------------------------------
3510 *YOMU
3520 '--------------------------------------------------------------
3530 OPEN "2:indx02.dat" FOR INPUT AS #3
3540 WHILE NOT EOF(3)
3550   COUNT=COUNT+1
3560   INPUT #3,DUMMY
3570 WEND
3580 CLOSE #3
3590 RETURN
3600 '
4000 '--------------------------------------------------------------
4010 *HIZUKE
4020 '--------------------------------------------------------------
4030 GOSUB *BUZZER
4040 LOCATE 27,19 : COLOR 6
4050 PRINT " ﾍﾞｰｽ ﾉ 年月日 ｦ ﾄﾞｳｿﾞ(ｾｲﾚｷ) "
4060 COLOR 0
4070 LOCATE 25,3 : INPUT "ﾅﾝ年";YY$
4080 IF LEN(YY$)<4 THEN BEEP : GOTO 4070
4090 LOCATE 25,5 : INPUT "ﾅﾝ月";MM$
4100 LOCATE 25,7 : INPUT "ﾅﾝ日";DD$
4110 GOSUB *BUZZER
4120 GOSUB *KESU1 : GOSUB *KESU2
4130 LOCATE 25,3 : PRINT "ﾍﾞｰｽ ﾉ 年月日 :";
4140 PRINT YY$"年 "MM$"月 "DD$"日"
4150 LOCATE 27,19 : COLOR 6
4160 PRINT "  ﾏﾁｶﾞｲ ﾅｲﾃﾞｽｶ ( y/n )?  "
4170 COLOR 0
4180 I$=INKEY$ : IF I$="" THEN GOTO 4180
4190 IF I$="Y" OR I$="y" THEN GOTO 4220
```

```
4200 IF I$="N" OR I$="n" THEN GOTO 4240
4210 BEEP : GOTO 4180
4220 YY=VAL(YY$) : MM=VAL(MM$) : DD=VAL(DD$)
4230 GOSUB *KESU1 : GOSUB *KESU2 : RETURN
4240 GOSUB *KESU1 : GOSUB *KESU2 : GOTO 4040
4250 '
4500 '----------------------------------------------------
4510 *KESU1
4520 '----------------------------------------------------
4530 FOR I=3 TO 16
4540   LOCATE 20,I : PRINT SPC(40)
4550 NEXT I
4560 RETURN
4570 '
5000 '----------------------------------------------------
5010 *KESU2
5020 '----------------------------------------------------
5030 LOCATE 20,19 : PRINT SPC(40)
5040 RETURN
5050 '
5500 '----------------------------------------------------
5510 *BUZZER
5520 '----------------------------------------------------
5530 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
5540 RETURN
5550 '
6000 '----------------------------------------------------
6010 *SHOKUGYO
6020 '----------------------------------------------------
6030 GOSUB *BUZZER
6040 GOSUB *KESU1
6050 LOCATE 30,19: COLOR 4
6060 PRINT " ｼｮｸｷﾞｮｳ ﾊﾞﾝｺﾞｳｦ ﾄﾞｳｿﾞ" : COLOR 0
6070 LOCATE 30,3 : PRINT "ｶｲｼｬ ﾔｸｲﾝ ·········1"
6080 LOCATE 30,5 : PRINT "ｻﾗﾘｰﾏﾝ ···········2"
6090 LOCATE 30,7 : PRINT "ｶﾞｸｾｲ ·············3"
6100 LOCATE 30,9 : PRINT "ｼｭﾌ ················4"
6110 LOCATE 30,11: PRINT "ｿﾉﾀ ················5"
6120 PF$=INKEY$ : IF PF$="" THEN GOTO 6120
6130 GOSUB *KESU1 : GOSUB *KESU2
6140 LOCATE 20,3 : PRINT "ｺｰﾄﾞ        ?";CD
6150 LOCATE 20,4 : PRINT "ﾈﾝﾚｲ        ?";AO
```

```
6160 LOCATE 20,5 : PRINT "ｾｲﾍﾞﾂ(m/f)? ";A1$
6170 LOCATE 20,6 : PRINT "ｼｮｸｷﾞｮｳ   ? ";PF$
6180 RETURN
6190 '
6500 '―――――――――――――――――――――――――――――――――――――
6510 *BUFCLEAR
6520 '―――――――――――――――――――――――――――――――――――――
6530 FOR I=REC2+1 TO 3
6540   LSET NM$(I)=""
6550   LSET PN$(I)=""
6560   LSET AD$(I)=""
6570   LSET BA$(I)=""
6580 NEXT I
6590 RETURN
```

2・5 データ修正のプログラム

この節では、住所変更、あるいは改姓などの場合のデータ修正をテーマとしていますが、**住所録の不要になった人はキャンセルし、そのコード番号を新規の人に割り当て、性別、年齢、職業などをこのプログラムで入力すれば、幅広く利用することができます。**

とくに、このプログラムでは書き込み人数を500人に限定していますので、500人を超えるような場合には、無効の人のデータの上に新規の人のそれを書き込んでいけば、効率よく使えると思います。

処理の流れ

プログラムの作成に入る前に、本節での処理がどのように流れるか画面を示して簡単に説明しておきましょう。

なお、これらの画面は実際の場合に比べてメッセージなどの表示部分の比率が画面全体に対して大きくなっていますが、イメージ的にはあまりズレがないことをおことわりしておきます。

修正作業は次の順序で行われます。

❶ プログラムをRUNするとコード入力画面になります(図1)。

コードに対して「9999」がキーインされると終りです。また該当コードがない場合は、

トウロク　シテ　アリマセン　！

のエラーメッセージが表示されます。

❷ 名前が表示され修正するかどうかの確認をうながします(図2)。

修正をしたくなければキャンセルで、コード入力からやりなおしになります。

❸ メッセージに従って各項目を修正します(図3)。

古いデータが表示されますから、そのままでよければリターンキーを押してください。

❹ 職業メニューについては前節と同じように番号を選びます(図4)。

すべての項目に対して修正が終ると登録するかどうかの確認がありますからYまたはNで答えてください。

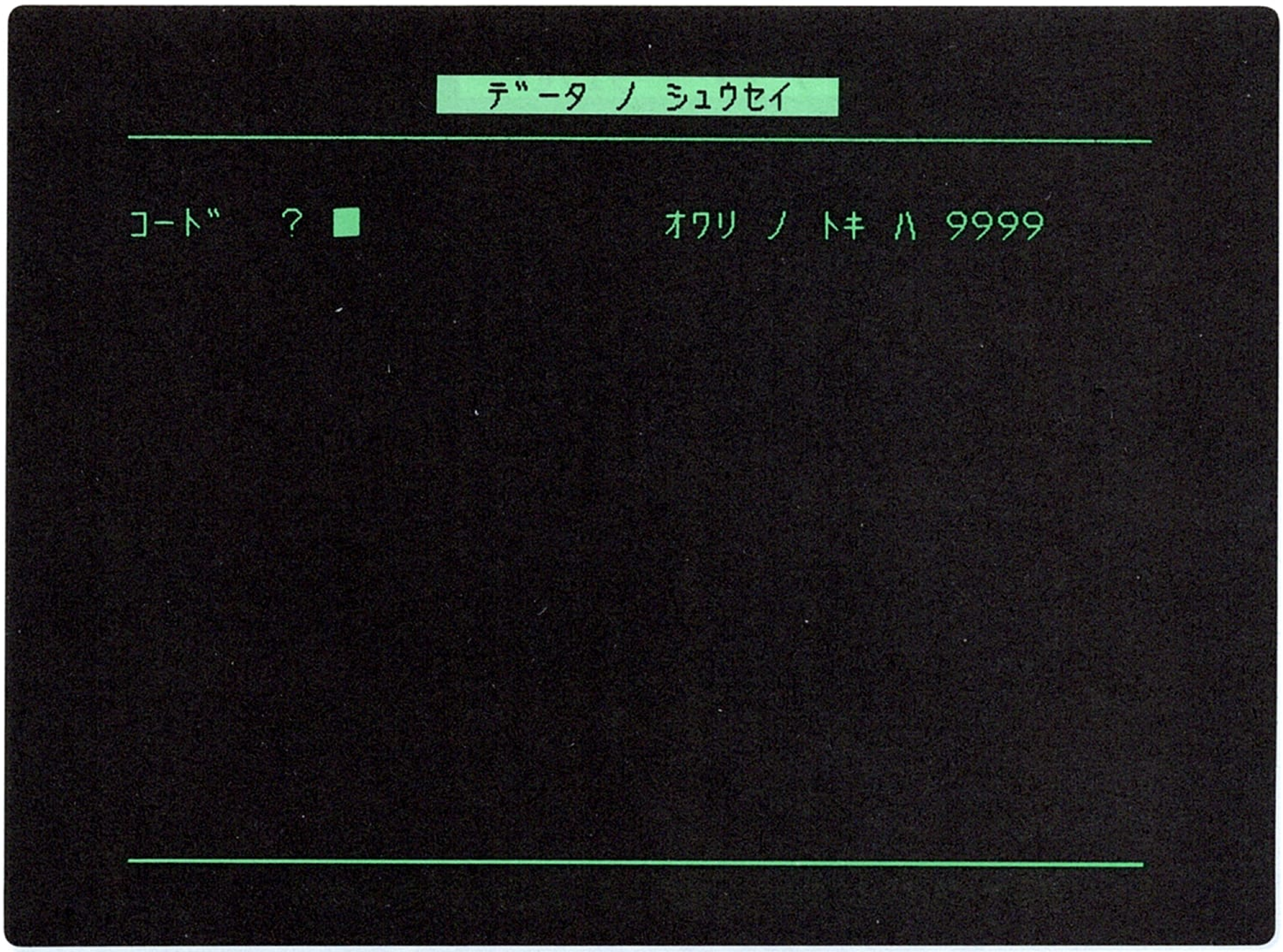


図1●コード入力画面

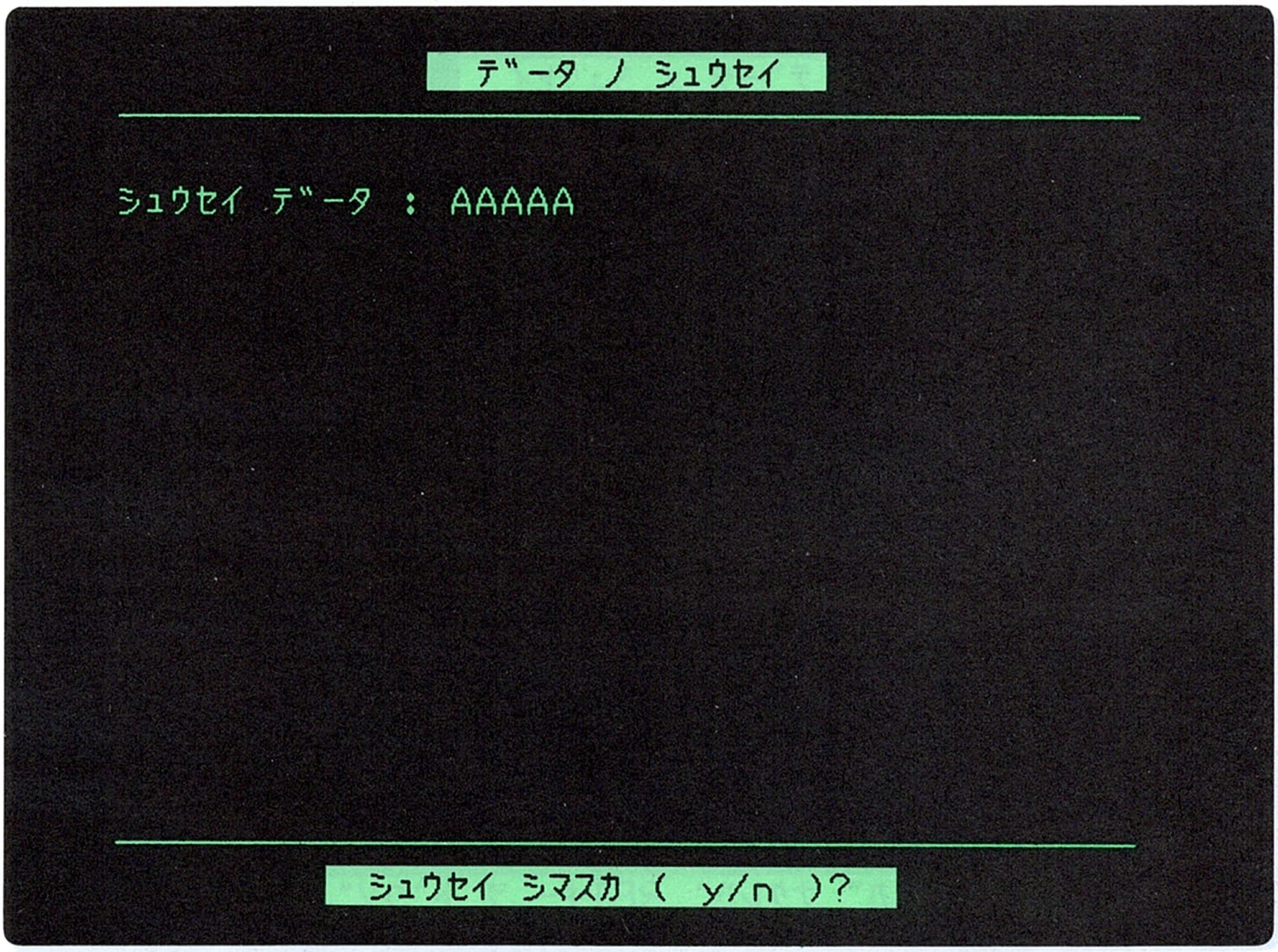


図2●修正データの確認

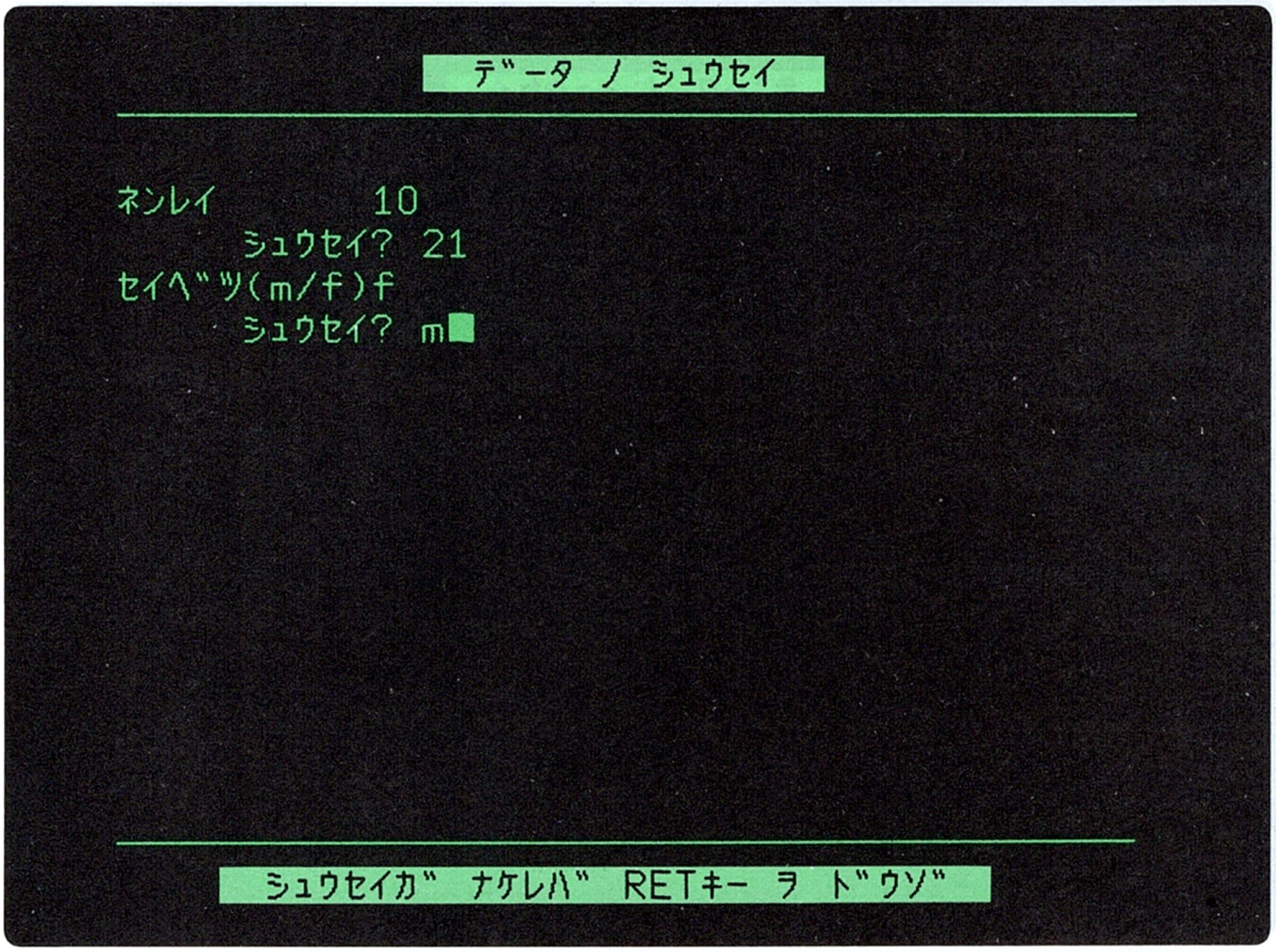


図3●データ修正中

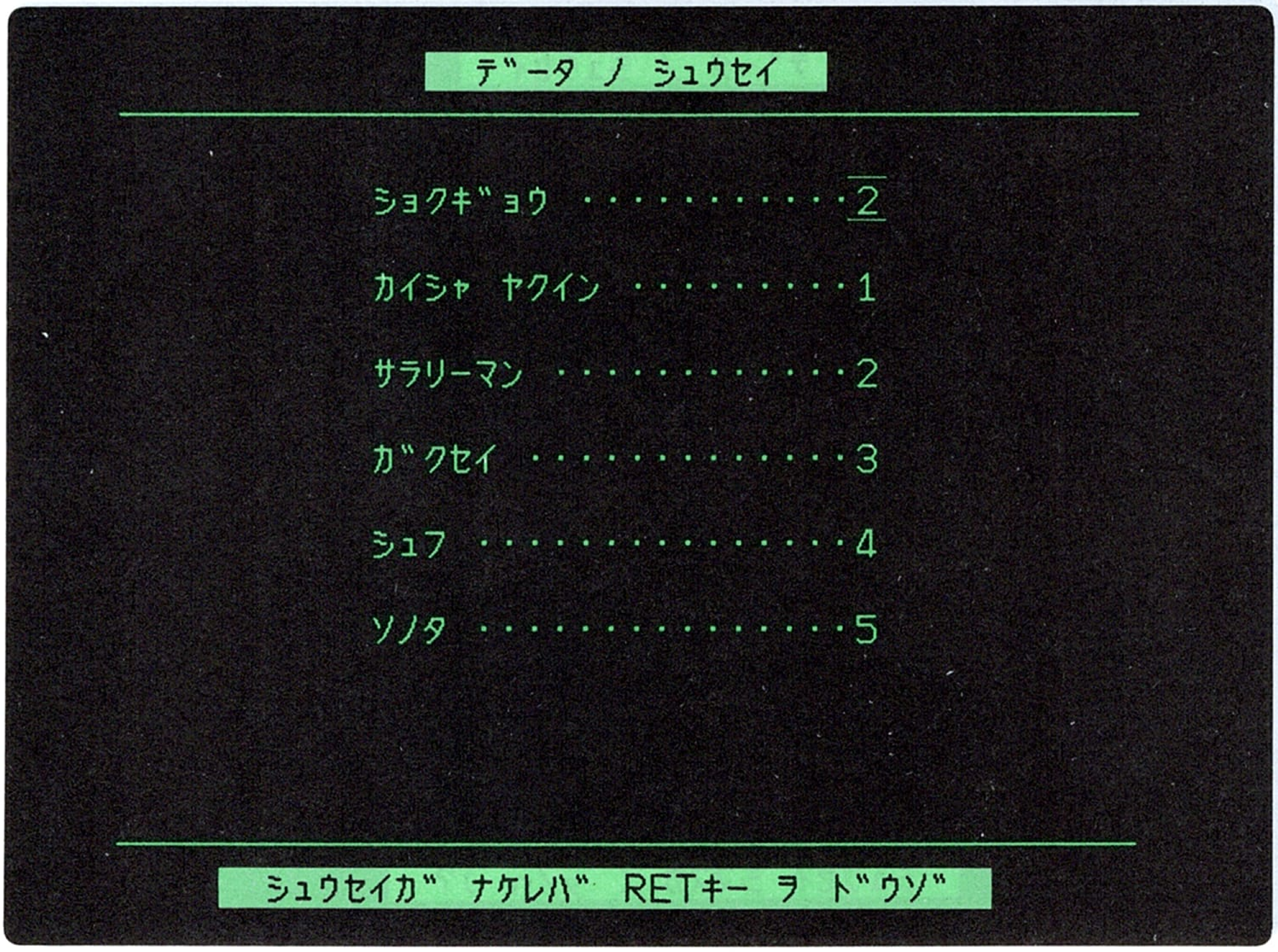


図4●職業メニュー画面

これからプログラムを作成していきますが、前節のそれとだいぶ似かよった点がありますので、説明は簡単にします。

初期設定からファイルオープンまで

次のプログラムを入力してください。

```
100 '  file name ····> DMAIL2.BAS
110 '  1985.2.15
120 '  ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ
130 '  save "DMAIL2.BAS"
140 '
150 '────────────────────────────────
160 *SHOKISETTEI
170 '────────────────────────────────
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DEF FNSEC1(X)=(X-1) ¥ 50+2
210 DEF FNREC1(X)=(X-1) MOD 50+1
220 DEF FNSEC2(X)=(X-1) ¥ 3+12
230 DEF FNREC2(X)=(X-1) MOD 3+1
240 DIM NM$(3),PN$(3),AD$(3),BA$(3)
250 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
260 '
270 '────────────────────────────────
280 *JYUNBI
290 '────────────────────────────────
300 GOSUB *FOPEN3
310 GET #1,1                         ← 第1セクタをバッファ上へ
320 OTOKO=CVI(Q2$) ┐
330 ONNA =CVI(Q3$) ┘                 ← 男女の数をカウンタに代入
340 CLOSE #1
350 GOSUB *FOPEN1 ┐
360 GOSUB *FOPEN2 ┘                  ← ランダムファイルをオープン
370 GOSUB *YOMU : DIM CD(COUNT) : COUNT=0
380 NIKAIME=-1 : GOSUB *YOMU         ← コードを配列に格納
390 '
400 LOCATE 30,0 : COLOR 4                              ┐
410 PRINT "      ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ      " : COLOR 0          │ 画面レイアウト
420 LOCATE 20,1 : PRINT STRING$(40,149)                │
430 LOCATE 20,18: PRINT STRING$(40,149)                ┘
440 '
```

次に**ファイルオープン**と**フィールド定義**のサブルーチンです。

```
2000 '―――――――――――――――――――――――――――
2010 *FOPEN1
2020 '―――――――――――――――――――――――――――
2030 OPEN "2:mail02.dat" AS #1
2040 FOR I=1 TO 50
2050   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
          2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
2060 NEXT I
2070 RETURN
2080 '
2500 '―――――――――――――――――――――――――――
2510 *FOPEN2
2520 '―――――――――――――――――――――――――――
2530 OPEN "2:mail02.dat" AS #2
2540 FOR I=1 TO 3
2550   FIELD #2,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
2560 NEXT I
2570 RETURN
2580 '
3000 '―――――――――――――――――――――――――――
3010 *FOPEN3
3020 '―――――――――――――――――――――――――――
3030 OPEN "2:mail02.dat" AS #1
3040 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
3050 RETURN
3060 '
3500 '―――――――――――――――――――――――――――
3510 *YOMU
3520 '―――――――――――――――――――――――――――
3530 OPEN "2:indx02.dat" FOR INPUT AS #3
3540 WHILE NOT EOF(3)
3550   COUNT=COUNT+1
3560   IF NIKAIME THEN GOTO 3580
3570   INPUT #3,DUMMY : GOTO 3590 ←―――データの読み飛ばし
3580   INPUT #3,CD(COUNT) ←―――――――コードを配列に格納
3590 WEND
3600 CLOSE #3
3610 RETURN
3620 '
```

コード入力とデータ確認

メインプログラムでコードを入力し、サブルーチンへジャンプしてから、修正データかどうかの確認をします。

```
450 '―――――――――――――――――――――――――――――――
460 *LOOP
470 '―――――――――――――――――――――――――――――――
480 LOCATE 40,3 : COLOR 6
490 PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  " : COLOR 0
500 LOCATE 20,3 : INPUT "ｺｰﾄﾞ";CD
510 IF CD=9999 THEN GOTO *OWARI
520 GOSUB *SAGASU                    ←データサーチサブルーチンへ
530 IF P=-1 THEN GOTO *LOOP          ←見つからなかったら初めから
540 IF NOT SHUUSEI THEN GOTO *LOOP   ←修正したくなければやりなおし
550 LOCATE 25,19 : COLOR 4
560 PRINT " ｼｭｳｾｲｶﾞ ﾅｹﾚﾊﾞ RETｷｰ ｦ ﾄﾞｳｿﾞ  "  ←オペレーションメッセージ
570 COLOR 0
580 '
```

```
4000 '―――――――――――――――――――――――――――――――
4010 *SAGASU
4020 '―――――――――――――――――――――――――――――――
4030 P=SEARCH(CD,CD)                  ←配列内にコードがあるか？
4040 IF P=-1 THEN GOSUB *MITOOROKU    ←なければエラーメッセージ
4050 SEC1=FNSEC1(P) : REC1=FNREC1(P)
4060 SEC2=FNSEC2(P) : REC2=FNREC2(P)
4070 GET #1,SEC1 : GET #2,SEC2        ←該当データのセクタをGET
4080 GOSUB *BUZZER                    ←ブザー音
4090 GOSUB *KESU1
4100 LOCATE 20,3 : PRINT "ｼｭｳｾｲ ﾃﾞｰﾀ : ";
4110 COLOR 4 : PRINT NM$(REC2) : COLOR 0  ←確認データ表示
4120 LOCATE 28,19 : COLOR 6
4130 PRINT "  ｼｭｳｾｲ ｼﾏｽｶ ( y/n )?  "
4140 COLOR 0
4150 I$=INKEY$ : IF I$="" THEN GOTO 4150
4160 IF I$="Y" OR I$="y" THEN GOTO 4190
4170 IF I$="N" OR I$="n" THEN GOTO 4200
4180 BEEP : BEEP : GOTO 4150
4190 SHUUSEI=-1                       ←変数SHUUSEIを「真」にする
4200 GOSUB *KESU1 : GOSUB *KESU2
4210 RETURN
4220 '
```

この部分も前回と同じですからわかると思います。リスト中のコメントを参照してください。

SEARCH関数で配列の各要素を探し、もし入力コードがなければ次のサブルーチンでエラーメッセージを表示することにします。

該当コードなし

```
4500 '──────────────────────────
4510 *MITOOROKU
4520 '──────────────────────────
4530 BEEP : BEEP : BEEP
4540 GOSUB *KESU1
4550 LOCATE 31,3 : COLOR 6
4560 PRINT "  ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ ! " : COLOR 0
4570 FOR I=1 TO 2000 : NEXT I
4580 GOSUB *KESU1
4590 RETURN 4210
4600 '
```

次に、**画面の部分消去**および**ブザー音**のサブルーチンをキーインしておきましょう。

```
5000 '──────────────────────────
5010 *KESU1
5020 '──────────────────────────
5030 FOR I=3 TO 16
5040   LOCATE 20,I : PRINT SPC(40)
5050 NEXT I
5060 RETURN
5070 '
5500 '──────────────────────────
5510 *KESU2
5520 '──────────────────────────
5530 LOCATE 20,19 : PRINT SPC(40)
5540 RETURN
5550 '
6000 '──────────────────────────
6010 *BUZZER
6020 '──────────────────────────
6030 FOR I=1 TO 300 : BEEP 1 : BEEP 0 : NEXT I
6040 RETURN
6050 '
```

データ修正のルーチン

コードを入力し、それをサーチして、もしそれが修正データであれば、このルーチンへ落ちてきて修正処理をします。

```
590 AGE=CVI(AGE$(REC1))
600 AGE$=MID$(STR$(AGE),2)
610 '
620 B1$=SEX$(REC1)
630 B2$=NM$(REC2)
640 B3$=PN$(REC2)
650 B4$=AD$(REC2)
660 B5$=BA$(REC2)
670 '
680 LOCATE 20,3 : PRINT "ﾈﾝﾚｲ        ";AGE$
690 LOCATE 20,4 : INPUT "      ｼｭｳｾｲ";A0
700 C0=A0
710 IF A0=0 THEN A0=AGE
720 LOCATE 20,5 : PRINT "ｾｲﾍﾞﾂ(m/f)";B1$
730 LOCATE 20,6 : INPUT "      ｼｭｳｾｲ";A1$
740 C1$=A1$
750 IF A1$="" THEN A1$=B1$         ELSE IF
    A1$="m" OR A1$="M" THEN MF=-1 ELSE FF=-1
760 GOSUB *SHOKUGYO
770 LOCATE 20,9 : PRINT "ﾅﾏｴ        ";B2$
780 LOCATE 20,10: INPUT "      ｼｭｳｾｲ";A2$
790 IF A2$="" THEN A2$=B2$
800 LOCATE 20,11: PRINT "ﾕｳﾋﾞﾝ No   ";B3$
810 LOCATE 20,12: INPUT "      ｼｭｳｾｲ";A3$
820 IF A3$="" THEN A3$=B3$
830 LOCATE 20,13: PRINT "ｼﾞｭｳｼｮ      ";B4$
840 LOCATE 20,14: INPUT "      ｼｭｳｾｲ";A4$
850 IF A4$="" THEN A4$=B4$
860 LOCATE 20,15: PRINT "ﾊﾞﾝﾁ        ";B5$
870 LOCATE 20,16: INPUT "      ｼｭｳｾｲ";A5$
880 IF A5$="" THEN A5$=B5$
890 GOSUB *BUZZER
900 '
```

修正方法は、最初、すでにディスクに書き込まれているデータを表示し、もし修正があればキーボードより入力し、そのままでよければリターンキーを押します。134ページの図3を見ていただけばわかる

と思います。

590～660行で、年齢、性別、名前、郵便番号、住所、番地などを新しい変数に代入していますが、これは、680行以降のプログラムを簡便にするための措置で、それ以上の目的はありません。

このように一度他の変数に代入するやり方は、MID＄とかLEFT＄などの関数を複数個用いて変換したりする場合に使います。あらかじめ変換しておいたほうが処理がすみやかに進むからです。

700行、740行で、Ｃ０およびＣ１＄に入力データを代入していますが、これは、760行で職業修正のサブルーチンへジャンプすると画面がクリアされますので、飛び先から帰ってきたときに、ふたたびいまのような画面を再現させるために古いデータを保存しておくのです。

750行は、新規のデータを入力したりして性別の変更があった場合に、**男女別のフラグを真にしておく処理**です。このフラグいかんによって、あとで男女数の増減をします。

このステートメントの意味は、**「もしリターンキーが押されていれば古いデータをそのまま生かすことになりELSE以下は実行しません。もしリターンキーでなかったらMかFが入力されたことになりますから、対応したフラグを立てる」**ということになります。

では次に、760行でのジャンプ先、職業メニューのサブルーチンです。

プログラムリストは右ページに示します。

この部分も説明なしでわかると思いますが、１つだけ注意しなければならないことがあります。それは6650行の**CHR＄(13)**です。

CHR＄(13)
●これはキャリッジリターンに対応するアスキーキャラクタです。

6630行でメニュー番号の入力待ちをしていますが、もしリターンキーの場合は変更なしなのでPF＄に古いデータを入れています。しかしこのときの条件式は、

IF PF$=" " THEN～

ではいけないのでしょうか？

ヌルストリング

リターンキーですから、PF＄には何も入っていない、つまり**ヌルストリング**と考えればいいように思えます。

これは一見正しいように見えますが、実は違うのです。右ページ下の画面のようなプログラムをテストしてみてください。

RUNさせるとキー入力待ちになりますから、リターンキーを押してみてください。画面に、

NULL

とは表示されず、**CR**と表示されました。つまりキャリッジリターン

```
6500 '―――――――――――――――――――――――――
6510 *SHOKUGYO
6520 '―――――――――――――――――――――――――
6530 GOSUB *BUZZER
6540 GOSUB *KESU1
6550 PRF$=PRF$(REC1)
6560 LOCATE 30,3 : PRINT "ショクギョウ ・・・・・・・・・・・・";
6570 COLOR 4 :PRINT PRF$ : COLOR 0
6580 LOCATE 30,5 : PRINT "カイシャ ヤクイン ・・・・・・・・・1"
6590 LOCATE 30,7 : PRINT "サラリーマン ・・・・・・・・・・・2"
6600 LOCATE 30,9 : PRINT "ガクセイ・・・・・・・・・・・・・・3"
6610 LOCATE 30,11: PRINT "シュフ ・・・・・・・・・・・・・・4"
6620 LOCATE 30,13: PRINT "ソノタ ・・・・・・・・・・・・・・5"
6630 PF$=INKEY$ : IF PF$="" THEN GOTO 6630
6640 C2$=PF$
6650 IF PF$=CHR$(13) THEN PF$=PRF$(REC1)
6660 GOSUB *KESU1 : GOSUB *KESU2
6670 LOCATE 20,3 : PRINT "ネンレイ          ";AGE$
6680 LOCATE 20,4 : PRINT "      シュウセイ?";C0
6690 LOCATE 20,5 : PRINT "セイベツ(m/f)";B1$
6700 LOCATE 20,6 : PRINT "      シュウセイ?";C1$
6710 LOCATE 20,7 : PRINT "ショクギョウ    ";PRF$
6720 LOCATE 20,8 : PRINT "      シュウセイ?";C2$
6730 RETURN
6740 '
```

```
list
100 I$=INKEY$ : IF I$="" THEN GOTO 100
110 IF I$="" THEN PRINT "NULL":GOTO 130
120 IF I$=CHR$(13) THEN PRINT "CR"
130 END
Ok
run
CR
Ok
█
```

の機能キャラクタと判断されたのです。

ですから、もし6650行の条件式にCHR＄(13)を使わずにヌルストリ

ングを使うと、リターンキーが押された場合（変更がないとき）この行にひっかかりませんから、書き込みの段階で古いデータが消されてしまいます。

INKEY $

キーバッファ

なお、**INKEY $**は、私たちがキーボードのキーを何か押すと必ず**キーバッファ**に取り込まれますが、それを1文字ずつ取り出す機能をもっています。

各項目の修正が終ると、次は登録するかどうかの確認です。

```
910 LOCATE 31,19 : COLOR 4
920 PRINT " ﾄｳﾛｸ ｼﾏｽｶ ( y/n )? ":COLOR 0
930 I$=INKEY$ : IF I$="" THEN GOTO 930
940 IF I$="Y" OR I$="y" THEN GOTO *KAKU
950 IF I$="N" OR I$="n" THEN GOTO *TORIKESHI
960 BEEP : BEEP : GOTO 930
970 '
```

データの書き込み

登録の場合は、ランダムファイルへの書き込みを実行し、取り消しの場合は、もう一度最初からやりなおしです。

登録のプログラムリストは次のとおりです。

```
1000 '────────────────────────────────
1010 *KAKU
1020 '────────────────────────────────
1030 IF MF THEN OTOKO=OTOKO+1:ONNA=ONNA-1 :MF=0
1040 IF FF THEN ONNA=ONNA+1:OTOKO=OTOKO-1 :FF=0
1050 LSET SEX$(REC1)=A1$            ┐
1060 LSET AGE$(REC1)=MKI$(A0)       ├─1番バッファでセット
1070 LSET PRF$(REC1)=PF$            ┘
1080 LSET NM$(REC2) =A2$  ┐
1090 LSET PN$(REC2) =A3$  ├─2番バッファでセット
1100 LSET AD$(REC2) =A4$  │
1110 LSET BA$(REC2) =A5$  ┘
1120 PUT #1,SEC1 : PUT #2,SEC2
1130 SHUUSEI=0 ←──────── 変数をクリア
1140 GOSUB *KESU1 : GOSUB *KESU2 : GOTO *LOOP
1150 '
```

変わったところというと、1030行、1040行で**男女数の変動があったときの調整**を行なっています。その判断規準は、先ほど立てておいたフラグですね。

このプログラムはデータ追加ではありませんから、データ総数は変わりません。ですから、たとえば男性が1人ふえれば女性は1人減らしておかなければなりませんね。

次は、**取り消しの場合の処理**です。画面を一部クリアして、コード入力へ戻します。

```
1160 '────────────────────────────
1170 *TORIKESHI
1180 '────────────────────────────
1190 GOSUB *KESU1 : GOSUB *KESU2
1200 SHUUSEI=0 : GOTO *LOOP
1210 '
```

終りの処理

コードに「9999」が入力されればファイルをいったんクローズし、第1セクタに男女数を書き込みます。そしてメニューを実行します。

〈注〉1270行で第1セクタをGET文でバッファ上へ取り出しておきませんと、日付、登録総数などがPUT文により消されてしまいます。

```
1220 '────────────────────────────
1230 *OWARI
1240 '────────────────────────────
1250 CLOSE
1260 GOSUB *FOPEN3
1270 GET #1,1
1280 LSET Q2$=MKI$(OTOKO)
1290 LSET Q3$=MKI$(ONNA)
1300 PUT #1,1 : CLOSE #1
1310 CLS
1320 RUN "menu2.bas"
1330 '
```

最後にプログラムをセーブしておしまいです。プログラム総リストを次ページ以降にかかげます。

```
save "DMAIL2.BAS" ⏎
```

```
100 ' file name ････> DMAIL2.BAS
110 ' 1985.2.15
120 ' ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ
130 ' save "DMAIL2.BAS"
140 '
150 '――――――――――――――――――――――――――――――――――――――――
160 *SHOKISETTEI
170 '――――――――――――――――――――――――――――――――――――――――
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DEF FNSEC1(X)=(X-1) ¥ 50+2
210 DEF FNREC1(X)=(X-1) MOD 50+1
220 DEF FNSEC2(X)=(X-1) ¥ 3+12
230 DEF FNREC2(X)=(X-1) MOD 3+1
240 DIM NM$(3),PN$(3),AD$(3),BA$(3)
250 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
260 '
270 '――――――――――――――――――――――――――――――――――――――――
280 *JYUNBI
290 '――――――――――――――――――――――――――――――――――――――――
300 GOSUB *FOPEN3
310 GET #1,1
320 OTOKO=CVI(Q2$)
330 ONNA =CVI(Q3$)
340 CLOSE #1
350 GOSUB *FOPEN1
360 GOSUB *FOPEN2
370 GOSUB *YOMU : DIM CD(COUNT) : COUNT=0
380 NIKAIME=-1 : GOSUB *YOMU
390 '
400 LOCATE 30,0 : COLOR 4
410 PRINT "    ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ    " : COLOR 0
420 LOCATE 20,1 : PRINT STRING$(40,149)
430 LOCATE 20,18: PRINT STRING$(40,149)
440 '
450 '――――――――――――――――――――――――――――――――――――――――
460 *LOOP
470 '――――――――――――――――――――――――――――――――――――――――
480 LOCATE 40,3 : COLOR 6
490 PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  " : COLOR 0
500 LOCATE 20,3 : INPUT "ｺｰﾄﾞ  ";CD
```

```
510 IF CD=9999 THEN GOTO *OWARI
520 GOSUB *SAGASU
530 IF P=-1 THEN GOTO *LOOP
540 IF NOT SHUUSEI THEN GOTO *LOOP
550 LOCATE 25,19 : COLOR 4
560 PRINT " ｼｭｳｾｲｶﾞ ﾅｹﾚﾊﾞ RETｷｰ ｦ ﾄﾞｳｿﾞ   "
570 COLOR 0
580 '
590 AGE=CVI(AGE$(REC1))
600 AGE$=MID$(STR$(AGE),2)
610 '
620 B1$=SEX$(REC1)
630 B2$=NM$(REC2)
640 B3$=PN$(REC2)
650 B4$=AD$(REC2)
660 B5$=BA$(REC2)
670 '
680 LOCATE 20,3 : PRINT "ﾈﾝﾚｲ        ";AGE$
690 LOCATE 20,4 : INPUT "      ｼｭｳｾｲ";A0
700 C0=A0
710 IF A0=0 THEN A0=AGE
720 LOCATE 20,5 : PRINT "ｾｲﾍﾞﾂ(m/f)";B1$
730 LOCATE 20,6 : INPUT "      ｼｭｳｾｲ";A1$
740 C1$=A1$
750 IF A1$="" THEN A1$=B1$         ELSE IF
    A1$="m" OR A1$="M" THEN MF=-1 ELSE FF=-1
760 GOSUB *SHOKUGYO
770 LOCATE 20,9 : PRINT "ﾅﾏｴ        ";B2$
780 LOCATE 20,10: INPUT "      ｼｭｳｾｲ";A2$
790 IF A2$="" THEN A2$=B2$
800 LOCATE 20,11: PRINT "ﾕｳﾋﾞﾝ No   ";B3$
810 LOCATE 20,12: INPUT "      ｼｭｳｾｲ";A3$
820 IF A3$="" THEN A3$=B3$
830 LOCATE 20,13: PRINT "ｼﾞｭｳｼｮ     ";B4$
840 LOCATE 20,14: INPUT "      ｼｭｳｾｲ";A4$
850 IF A4$="" THEN A4$=B4$
860 LOCATE 20,15: PRINT "ﾊﾞﾝﾁ       ";B5$
870 LOCATE 20,16: INPUT "      ｼｭｳｾｲ";A5$
880 IF A5$="" THEN A5$=B5$
890 GOSUB *BUZZER
900 '
```

```
910 LOCATE 31,19 : COLOR 4
920 PRINT " ﾄｳﾛｸ ｼﾏｽｶ ( y/n )? ":COLOR 0
930 I$=INKEY$ : IF I$="" THEN GOTO 930
940 IF I$="Y" OR I$="y" THEN GOTO *KAKU
950 IF I$="N" OR I$="n" THEN GOTO *TORIKESHI
960 BEEP : BEEP : GOTO 930
970 '
1000 '────────────────────────────────────
1010 *KAKU
1020 '────────────────────────────────────
1030 IF MF THEN OTOKO=OTOKO+1:ONNA=ONNA-1 :MF=0
1040 IF FF THEN ONNA=ONNA+1:OTOKO=OTOKO-1 :FF=0
1050 LSET SEX$(REC1)=A1$
1060 LSET AGE$(REC1)=MKI$(A0)
1070 LSET PRF$(REC1)=PF$
1080 LSET NM$(REC2) =A2$
1090 LSET PN$(REC2) =A3$
1100 LSET AD$(REC2) =A4$
1110 LSET BA$(REC2) =A5$
1120 PUT #1,SEC1 : PUT #2,SEC2
1130 SHUUSEI=0
1140 GOSUB *KESU1 : GOSUB *KESU2 : GOTO *LOOP
1150 '
1160 '────────────────────────────────────
1170 *TORIKESHI
1180 '────────────────────────────────────
1190 GOSUB *KESU1 : GOSUB *KESU2
1200 SHUUSEI=0 : GOTO *LOOP
1210 '
1220 '────────────────────────────────────
1230 *OWARI
1240 '────────────────────────────────────
1250 CLOSE
1260 GOSUB *FOPEN3
1270 GET #1,1
1280 LSET Q2$=MKI$(OTOKO)
1290 LSET Q3$=MKI$(ONNA)
1300 PUT #1,1 : CLOSE #1
1310 CLS
1320 RUN "menu2.bas"
1330 '
```

```
2000 '────────────────────────────────────────
2010 *FOPEN1
2020 '────────────────────────────────────────
2030 OPEN "2:mail02.dat" AS #1
2040 FOR I=1 TO 50
2050   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
         2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
2060 NEXT I
2070 RETURN
2080 '
2500 '────────────────────────────────────────
2510 *FOPEN2
2520 '────────────────────────────────────────
2530 OPEN "2:mail02.dat" AS #2
2540 FOR I=1 TO 3
2550   FIELD #2,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
2560 NEXT I
2570 RETURN
2580 '
3000 '────────────────────────────────────────
3010 *FOPEN3
3020 '────────────────────────────────────────
3030 OPEN "2:mail02.dat" AS #1
3040 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
               2 AS Q4$,2 AS Q5$,2 AS Q6$
3050 RETURN
3060 '
3500 '────────────────────────────────────────
3510 *YOMU
3520 '────────────────────────────────────────
3530 OPEN "2:indx02.dat" FOR INPUT AS #3
3540 WHILE NOT EOF(3)
3550   COUNT=COUNT+1
3560   IF NIKAIME THEN GOTO 3580
3570   INPUT #3,DUMMY : GOTO 3590
3580   INPUT #3,CD(COUNT)
3590 WEND
3600 CLOSE #3
3610 RETURN
3620 '
```

```
4000 '────────────────────────────────────────────
4010 *SAGASU
4020 '────────────────────────────────────────────
4030 P=SEARCH(CD,CD)
4040 IF P=-1 THEN GOSUB *MITOOROKU
4050 SEC1=FNSEC1(P) : REC1=FNREC1(P)
4060 SEC2=FNSEC2(P) : REC2=FNREC2(P)
4070 GET #1,SEC1 : GET #2,SEC2
4080 GOSUB *BUZZER
4090 GOSUB *KESU1
4100 LOCATE 20,3 : PRINT "ｼｭｳｾｲ ﾃﾞｰﾀ : ";
4110 COLOR 4 : PRINT NM$(REC2) : COLOR 0
4120 LOCATE 28,19 : COLOR 6
4130 PRINT "  ｼｭｳｾｲ ｼﾏｽｶ ( y/n )?  "
4140 COLOR 0
4150 I$=INKEY$ : IF I$="" THEN GOTO 4150
4160 IF I$="Y" OR I$="y" THEN GOTO 4190
4170 IF I$="N" OR I$="n" THEN GOTO 4200
4180 BEEP : BEEP : GOTO 4150
4190 SHUUSEI=-1
4200 GOSUB *KESU1 : GOSUB *KESU2
4210 RETURN
4220 '
4500 '────────────────────────────────────────────
4510 *MITOOROKU
4520 '────────────────────────────────────────────
4530 BEEP : BEEP : BEEP
4540 GOSUB *KESU1
4550 LOCATE 31,3 : COLOR 6
4560 PRINT "  ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ ! " : COLOR 0
4570 FOR I=1 TO 2000 : NEXT I
4580 GOSUB *KESU1
4590 RETURN 4210
4600 '
5000 '────────────────────────────────────────────
5010 *KESU1
5020 '────────────────────────────────────────────
5030 FOR I=3 TO 16
5040   LOCATE 20,I : PRINT SPC(40)
5050 NEXT I
5060 RETURN
```

```
5070 '
5500 '──────────────────────────────────────────
5510 *KESU2
5520 '──────────────────────────────────────────
5530 LOCATE 20,19 : PRINT SPC(40)
5540 RETURN
5550 '
6000 '──────────────────────────────────────────
6010 *BUZZER
6020 '──────────────────────────────────────────
6030 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
6040 RETURN
6050 '
6500 '──────────────────────────────────────────
6510 *SHOKUGYO
6520 '──────────────────────────────────────────
6530 GOSUB *BUZZER
6540 GOSUB *KESU1
6550 PRF$=PRF$(REC1)
6560 LOCATE 30,3 : PRINT "ｼｮｸｷﾞｮｳ ･･･････････";
6570 COLOR 4 :PRINT PRF$ : COLOR 0
6580 LOCATE 30,5 : PRINT "ｶｲｼｬ ﾔｸｲﾝ ････････1"
6590 LOCATE 30,7 : PRINT "ｻﾗﾘｰﾏﾝ ････････････2"
6600 LOCATE 30,9 : PRINT "ｶﾞｸｾｲ ･････････････3"
6610 LOCATE 30,11: PRINT "ｼｭﾌ ･･･････････････4"
6620 LOCATE 30,13: PRINT "ｿﾉﾀ ･･･････････････5"
6630 PF$=INKEY$ : IF PF$="" THEN GOTO 6630
6640 C2$=PF$
6650 IF PF$=CHR$(13) THEN PF$=PRF$(REC1)
6660 GOSUB *KESU1 : GOSUB *KESU2
6670 LOCATE 20,3 : PRINT "ﾈﾝﾚｲ        ";AGE$
6680 LOCATE 20,4 : PRINT "      ｼｭｳｾｲ?";C0
6690 LOCATE 20,5 : PRINT "ｾｲﾍﾞﾂ(m/f)";B1$
6700 LOCATE 20,6 : PRINT "      ｼｭｳｾｲ?";C1$
6710 LOCATE 20,7 : PRINT "ｼｮｸｷﾞｮｳ    ";PRF$
6720 LOCATE 20,8 : PRINT "      ｼｭｳｾｲ?";C2$
6730 RETURN
6740 '
```

2·6 プログラムの実行と確認

さて、それではRUNさせてみましょう。正常に走るかどうかチェックするデータとして、122ページの７番目の人を新規の人にすることにして次のようにしてみます。コードは変わりませんよ。

コード——**104**	氏　名——**ABCDE**
年　齢——**21**	郵便番号—**123**
性　別——**f**	住　所——**abcdefghij**
職　業——**3**	番　地——**1—2—3**

準備はいいですか？

さあ、f・5 キーを押してみてください。

すると、画面はコード入力となります。

コード「104」を入力しリターンキーを押すと、「GGGGG」さんでよいかどうか聞いてきますから、「Y」を入力してください。

続いてデータ修正の画面になりますから、上に示したデータに変更してください。この過程を示す画面は次のようになります。

```
            デ゛ータ ノ シュウセイ
────────────────────────────────────

ネンレイ        32
      シュウセイ? 21
セイヘ゛ツ(m/f)m
      シュウセイ?f
ショクキ゛ョウ   2
      シュウセイ?3
ナマエ          GGGGG
      シュウセイ? ABCDE
ユウヒ゛ン No   777
      シュウセイ? 123
シ゛ュウショ     gggggggggg
      シュウセイ? abcdefghi
ハ゛ンチ        1-4
      シュウセイ? 1-2-3

────────────────────────────────────
      トウロク シマスカ ( y/n )?
```

修正データを入れると左ページの画面のように確認メッセージが現れますから、よければ「Y」を入力してください。そしてここでは、1名の修正で処理を終りにします。

次にデータが正しく修正されたかどうか確認してみたいと思いますが、123ページでセーブしたテスト用のプログラムをロードしてください。

```
load "DMTEST"⏎
```

ロードできたらこのプログラムをRUNさせます。「Ok」が表示されたらダイレクトモードでGETして調べてみてください。次に入力例を示しますが、この画面では修正データだけを確認しました。最後にCLOSEするのをお忘れなく。

```
get #1,2
Ok
? sex$(7);cvi(age$(7));prf$(7)
f 21 3  ←――― 性別、年齢、職種
Ok
get #2,14
Ok
? nm$(1):? pn$(1):? ad$(1):? ba$(1)
ABCDE       ┐
123         │
abcdefghij  ├――― 氏名、郵便番号、住所、番地
1-2-3       ┘
Ok
get #3,1
Ok
? cvi(q1$)cvi(q2$)cvi(q3$)
 7  3  4  ←――― 女性が1人ふえた
    ↑
Ok  └――――――― 男性が1人減った
close
Ok
■
```

2・7　データ更新のプログラム

データの新規書き込み、データ修正のプログラムがかたずきました。どうですか？　プログラミングというのは、やってみるとけっこうおもしろいと思いませんか？

エラーメッセージがでたり、正しいデータが得られなかったりしたとき、その原因を探すのも案外楽しいものです。

そして、そんなことをしているうちに、思いがけないテクニックに気がつき、だんだんと体得できていくものなのです。

ディスクの扱いにしても、前編では、ごく当たりまえの方法でしかデータの出し入れをしませんでしたが、この本を読んでみて、型にはめずに、自由に扱えることが徐々におわかりいただけたと思います。そして、この、制限にとらわれないメリットがあるからこそ、ランダムファイルなのです。

この本で、いろいろな方法をマスターし、あなたの用途・目的にあったプログラム作りをするよう努力してください。

前おきはこのくらいにして、本節では、すでに書き込んであるデータを新しくする方法を学びます。

前節の修正プログラムもデータを改めることに変わりありませんが、ここでは、**数値データを書き換え**てみます。

私たちがこれまでに書き込んだランダムファイルには、何人かの人の年齢が保存されています。人は時間がたてば年をとるのは自然の摂理、このランダムファイルに登録してある人も例外ではありません。年はとりたくないけど、こればかりはどうにもなりません。

とすれば、私たちのデータも、1年ごとに更新していかなければ、データの意味が失われてしまいますね。

そこで、毎年4月1日に1歳ずつくり上げることにします。普通は誕生日がきたら、ということになりますが、ここでは、どの人も4月1日を誕生日にしてしまいましょう。

高校生、大学生、30歳未満の社会人など、年齢制限によりダイレクトメールを発送する場合、いま述べたくらいの基準でも、それほど狂いはないと思います。

プログラムはきわめて単純なものですが、ランダムファイルでの数値データの扱いの基本が含まれていますので、しっかりマスターするよう心がけてください。

すでに書き込まれているデータはストリングですので、これを数値に変更し、それに1を加えた結果を再びもとの位置に書き込めばよいのです。プログラムの流れは、一口にいうとこんなふうに要約できるかと思います。

初期設定

この部分は、前節までとほとんど変わりません。プログラムを次に示します。

```
100 '  file name ････> DMAIL3.BAS
110 '  1985.2.15
120 '  ﾃﾞｰﾀ ﾉ ｺｳｼﾝ
130 '  save "DMAIL3.BAS"
140 '
150 '──────────────────────────────────
160 *SHOKISETTEI
170 '──────────────────────────────────
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
210 DIM AGE(50)
220 '      └──────────更新した年齢を一時保存する配列
```

くれぐれもファイル名は間違わないようにしてください。私はひどい目にあった経験があります。

行番号310の配列AGEは、コメントにあるように、前のデータに1だけ加算した値を一時的に保存しておくためのものです。

また200行で

AGE$

という配列を、そして210行で

AGE

という配列を使用していますが、スペルが同じでも$マークがつくかつかないかで、まったく異なった配列になります。前者は**文字型（ストリング）配列**、後者は**整数型配列**です。

文字型配列

整数型配列

このように似かよった配列あるいは変数名を使いますと、プログラム作成中によく間違いますので注意してください。

前回更新日の確認

１年に１度更新するのでしたら間違わないと思いますが、このようなこともできるということを知っていただくために入れておきました。

プログラムを次に示します。

```
230 '―――――――――――――――――――――――――――――
240 *JYUNBI
250 '―――――――――――――――――――――――――――――
260 GOSUB *FOPEN2
270 GET #1,1
280 TOTAL=CVI(Q1$)
290 YY=CVI(Q4$) : MM=CVI(Q5$) : DD=CVI(Q6$)
300 KK=YY+1 : CLOSE #1
310 KK$=MID$(STR$(KK),2)
320 YY$=MID$(STR$(YY),2)
330 MM$=MID$(STR$(MM),2)
340 DD$=MID$(STR$(DD),2)
350 '
360 LOCATE 30,0 : COLOR 4
370 PRINT "     ﾃﾞｰﾀ ﾉ ｺｳｼﾝ          " : COLOR 0
380 LOCATE 20,1 : PRINT STRING$(40,149)
390 LOCATE 20,7 : PRINT STRING$(40,149)
400 LOCATE 25,3
410 PRINT YY$;"年";MM$;"月";DD$;"日 ｷ ｦ"
420 LOCATE 30,5
430 PRINT KK$;"年";MM$;"月";DD$;"日 ｷ ﾆ ｺｳｼﾝ"
440 '
450 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
460 '
470 LOCATE 30,8 : COLOR 6
480 PRINT " ｺｳｼﾝ ｼﾏｽｶ ( y/n )?  "
490 COLOR 0
500 I$=INKEY$ : IF I$="" THEN GOTO 500
510 IF I$="Y" OR I$="y" THEN GOTO *KASAN
520 IF I$="N" OR I$="n" THEN GOTO *OWARI
530 '
```

まずランダムファイルをOPENし、第１セクタをGETし、そこに書き込んであった年月日を取り出して文字型に変換します。これは符号を取り除いて出力を整えるためです。

300行の変数KKですが、これは430行のKK＄を見ればわかるように、

今期の西暦を表示するために使います。

第2章で作っている一連のプログラムでは、ランダムファイルの第1セクタの頭から**登録データ数、男性の登録数、女性の登録数、年、月、日**と順に、**各2バイトずつ**使っていました。

そして、年月日は書き込みをするときいつも入力していましたが、ベースとなる年月日は一度だけでよいわけですから、ある年度の最初には期首年月日とデータ入力日を、その年度内でのデータ追加のときはデータ入力日だけを、ということにして2つの日付を第1セクタに書き込むようにすると、もっと意味がでてくるかもしれません。

第1セクタはまだまだたくさんアキがあるのですから、有効な使い方をみなさん考えてみてください。

280行で**登録データ数**を数値に変換してメモリ上に呼び出していますが、これはあとで**GETするセクタ計算に使います。**

260行にGOSUB文がありますので、ファイルOPENとFIELD設定のサブルーチンをキーインしておきましょう。

いつもとまったく同じです。

```
2000 '―――――――――――――――――――――――――――――――――
2010 *FOPEN2
2020 '―――――――――――――――――――――――――――――――――
2030 OPEN "2:mail02.dat" AS #1
2040 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
            2 AS Q4$,2 AS Q5$,2 AS Q6$
2050 RETURN
```

もう1つ、性別、年齢などのフィールド設定をする必要がありますので、次のサブルーチンもキーインしておきます。

```
1000 '―――――――――――――――――――――――――――――――――
1010 *FOPEN1
1020 '―――――――――――――――――――――――――――――――――
1030 OPEN "2:mail02.dat" AS #1
1040 FOR I=1 TO 50
1050   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
         2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
1060 NEXT I
1070 RETURN
1080 '
```

年齢加算と書き込み部分

すでに書き込まれている年齢を数値変換してメモリ上に取り出し、1だけ加算して再びPUTするルーチンです。プログラムを次に示します。

```
540 '─────────────────────────────────
550 *KASAN
560 '─────────────────────────────────
570 YY=YY+1 ←──────────── 期首年も更新
580 GOSUB *FOPEN1
590 S.LENGTH=(TOTAL-1) ¥ 50+2 ←──── データが書き込まれている
                                      最終セクタを計算
600 FOR I=2 TO S.LENGTH
610   GET #1,I
620   FOR J=1 TO 50
630     C=C+1 ←──────────── カウンタ
640     IF C>TOTAL THEN GOTO 680 ←── カウンタが登録総数を超えたら
                                      強制的にループから抜ける
650     AGE(J)=CVI(AGE$(J))+1
660     LSET AGE$(J)=MKI$(AGE(J)) ←── 更新データをLSET
670   NEXT J
680   PUT #1,I ←──────────── データの書き込み
690   FOR J=1 TO 50 : AGE(J)=0 : NEXT J ←┐
700 NEXT I                               │
710 CLOSE #1              配列AGEをクリア─┘
720 '
```

使用セクタの計算

●普通、使用セクタを知るにはLOF関数を使いますが、この例では12セクタ以降も異なるフォーマットで使用していますので、その手前の第2～第11セクタのどこまで使用しているかということはLOF関数ではわかりません。

570行では、期首年もいっしょにプラス1してしまいました。

590行は、第2セクタから第11セクタまでのどのセクタまでデータが書き込まれているか計算するステートメントです。これをやっておかないと、第11セクタまでGETすることになり、だいぶムダな仕事をすることになってしまいます。

原理は、**初項を2として、50レコード経過するたびにセクタを1進める**にはどう計算すればよいか、ということです。

この種の計算は、前にセクタ番号、そのセクタ内でのレコード番号を計算するときにやりましたのでわかりますね。

630行のカウンタCは何のために入れてあるかといいますと、普通は620行～670行のJのループで50回仕事が行われますが、たとえば登録総数が87などという場合、2回目にJのループへ来たとき、まるまる50回ループするまでのことはなく、37回でループから抜け出してよいわけです。ですから、カウンタCで加算していって、これが登録総

数よりも大きくなったときに仕事をやめてループを抜け出させればよいわけです。

650行で年齢を1ずつ加算しています。そして、その加算された新しい年齢データを660行でLSETしています。これを、ループから抜けたときにセクタ単位でPUTすれば年齢が更新されますね。

690行は、配列AGEの内容をゼロクリアするためですが、このプログラムでは必要ないと思います。**一度使命を果たした変数はクリアしておく**、という原則を習慣づけるためにいれておきました。

年齢の更新が終りましたら、今度は第1セクタへの年月日の書き込みです。ランダムファイルは閉じられていますから、もう一度オープンしなければなりませんね。

そして最後にメニュープログラムの実行です。

```
730 '―――――――――――――――――――――――――――
740 *OWARI
750 '―――――――――――――――――――――――――――
760 CLOSE
770 GOSUB *FOPEN2
780 GET #1,1
790 LSET Q4$=MKI$(YY)
800 LSET Q5$=MKI$(MM)
810 LSET Q6$=MKI$(DD)
820 PUT #1,1 : CLOSE #1
830 CLS
840 RUN "menu2.bas"
850 '
```

では、**このプログラムをセーブ**して終りです。

```
save "DMAIL3.BAS" ↵
```

このプログラムの実行は次節で行います。

次ページにプログラム総リストをかかげます。

```
100 '  file name ････> DMAIL3.BAS
110 '  1985.2.15
120 '  ﾃﾞｰﾀ ﾉ ｺｳｼﾝ
130 '  save "DMAIL3.BAS"
140 '
150 '--------------------------------------------------------
160 *SHOKISETTEI
170 '--------------------------------------------------------
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
210 DIM AGE(50)
220 '
230 '--------------------------------------------------------
240 *JYUNBI
250 '--------------------------------------------------------
260 GOSUB *FOPEN2
270 GET #1,1
280 TOTAL=CVI(Q1$)
290 YY=CVI(Q4$) : MM=CVI(Q5$) : DD=CVI(Q6$)
300 KK=YY+1 : CLOSE #1
310 KK$=MID$(STR$(KK),2)
320 YY$=MID$(STR$(YY),2)
330 MM$=MID$(STR$(MM),2)
340 DD$=MID$(STR$(DD),2)
350 '
360 LOCATE 30,0 : COLOR 4
370 PRINT "     ﾃﾞｰﾀ ﾉ ｺｳｼﾝ      " : COLOR 0
380 LOCATE 20,1 : PRINT STRING$(40,149)
390 LOCATE 20,7 : PRINT STRING$(40,149)
400 LOCATE 25,3
410 PRINT YY$;"年";MM$;"月";DD$;"日 ｷ ｦ"
420 LOCATE 30,5
430 PRINT KK$;"年";MM$;"月";DD$;"日 ｷ ﾆ ｺｳｼﾝ"
440 '
450 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
460 '
470 LOCATE 30,8 : COLOR 6
480 PRINT " ｺｳｼﾝ ｼﾏｽｶ ( y/n )? "
490 COLOR 0
500 I$=INKEY$ : IF I$="" THEN GOTO 500
```

```
510 IF I$="Y" OR I$="y" THEN GOTO *KASAN
520 IF I$="N" OR I$="n" THEN GOTO *OWARI
530 '
540 '―――――――――――――――――――――――――――
550 *KASAN
560 '―――――――――――――――――――――――――――
570 YY=YY+1
580 GOSUB *FOPEN1
590 S.LENGTH=(TOTAL-1) ¥ 50+2
600 FOR I=2 TO S.LENGTH
610   GET #1,I
620   FOR J=1 TO 50
630     C=C+1
640     IF C>TOTAL THEN GOTO 680
650     AGE(J)=CVI(AGE$(J))+1
660     LSET AGE$(J)=MKI$(AGE(J))
670   NEXT J
680   PUT #1,I
690   FOR J=1 TO 50 : AGE(J)=0 : NEXT J
700 NEXT I
710 CLOSE #1
720 '
730 '―――――――――――――――――――――――――――
740 *OWARI
750 '―――――――――――――――――――――――――――
760 CLOSE
770 GOSUB *FOPEN2
780 GET #1,1
790 LSET Q4$=MKI$(YY)
800 LSET Q5$=MKI$(MM)
810 LSET Q6$=MKI$(DD)
820 PUT #1,1 : CLOSE #1
830 CLS
840 RUN "menu2.bas"
850 '
1000 '―――――――――――――――――――――――――――
1010 *FOPEN1
1020 '―――――――――――――――――――――――――――
1030 OPEN "2:mail02.dat" AS #1
1040 FOR I=1 TO 50
```

```
1050   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
          2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
1060 NEXT I
1070 RETURN
1080 '
2000 '________________________________________
2010 *FOPEN2
2020 '________________________________________
2030 OPEN "2:mail02.dat" AS #1
2040 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
2050 RETURN
```

2·8　データは書き換えられているか？

さあ、それではさっそく実行して、プログラムが正常に動くかどうか確認してみましょう。

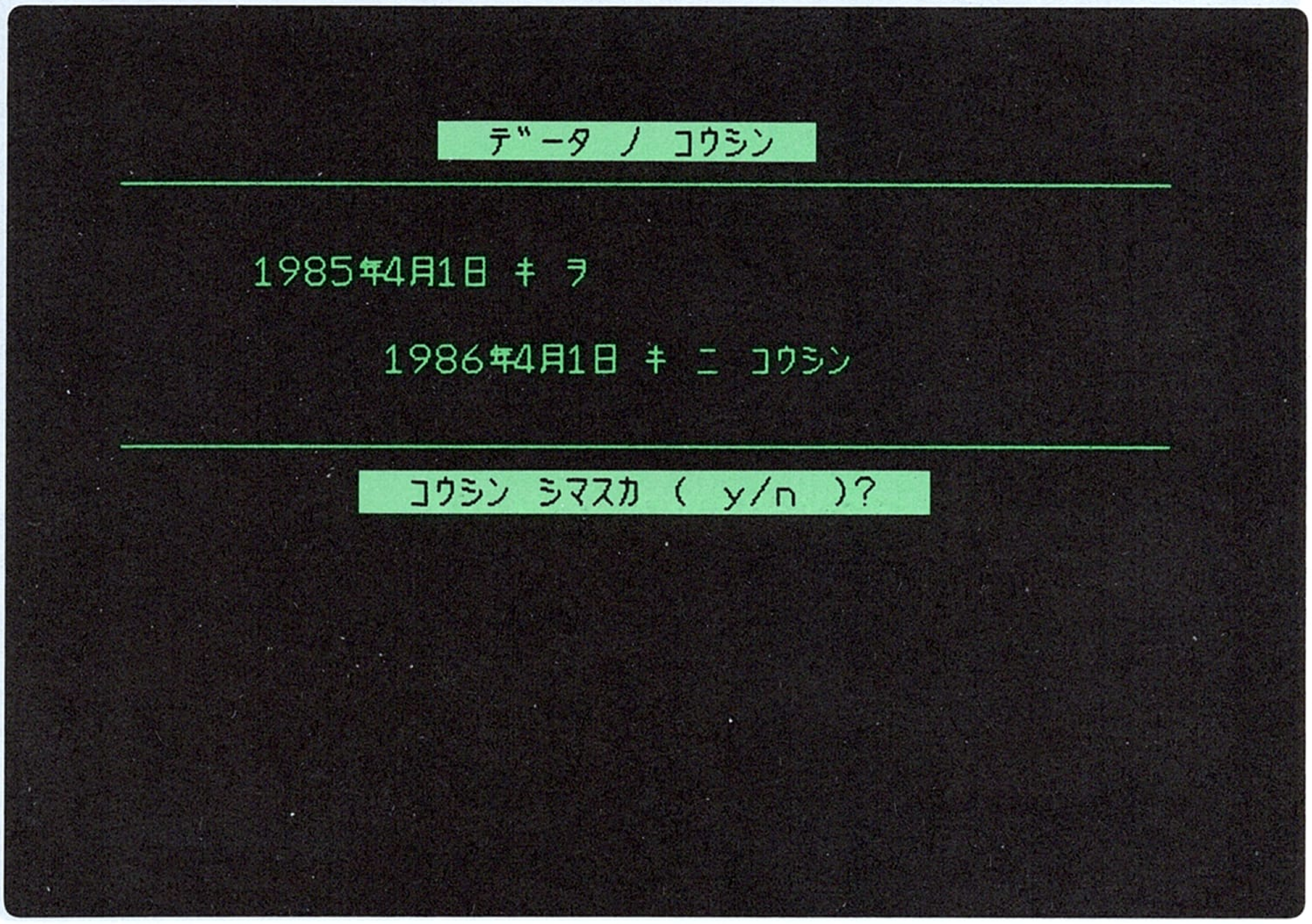
プログラムがロードできていない方は

DMAIL3.BAS

をロードしてください。

そして f・5 キーを押してください。

すると次のような確認の画面になります。



私たちは1985年をベースの年にしていましたので、1986年度期に切り換えるわけですね。

Yを入力してみましょう。

すると、またたく間に書き換えが行われ、その後メニュープログラムを実行しようとしますが、まだ作っていないため、

File not found

のエラーメッセージが表示されます。

これは余り気にしないで次へ進みましょう。つまり、本当に書き込

まれたかどうかをダイレクトモードでテストしてみるのです。

では**テストプログラムをロード**してください。

```
load "DMTEST" ⏎
```

そしてRUNさせ、次の画面のようにダイレクトモードで入力し確認してください。

```
run ←―――――――――――――――― テストプログラムの実行
Ok
get #1,2
Ok
for i=1 to 7:? cvi(age$(i)):next
 43 ┐
 11 │
 51 │
 28 ├―――――― 更新データ
 31 │        ●122ページのデータと比較してみてください
 20 │
 22 ┘
Ok
get #3,1
Ok
? cvi(q4$):? cvi(q5$):? cvi(q6$)
 1986 ←―――――――――――――― 年度も更新されている
 4
 1
Ok
close ←――――――――――――――― これを忘れないように
Ok
█
```

成功したようです。あなたはどうでしたか？

もしうまくいきませんでしたら、上の画面の**GETするバッファ番号、セクタが違っていないか**、あるいは**プログラム中のファイル番号が違っていないか**などをチェックしてください。

私たちがランダムファイルに書き込んだデータはテスト用です。うまくいかなかったためにもう一度更新プログラムを実行するにしても、わざわざデータを書き込みプログラムによって入力しなおす必要はないでしょう。加算が確認できればかまいません。

2・9 さらに汎用性のある宛て名印刷

私たちはこの章で、第１章の場合よりさらに項目をふやしてデータファイルを作ってきました。

そこで、まとめとして、いろいろな条件別に宛て名を出力するプログラムを作ってみましょう。出力方法は、プリンタをお持ちでない方もいると思いますので、第１章と同じように、ディスプレイにも出力できるようにしておきましょう。

6つの印刷メニュー

さて、いったいこのプログラムを走らせると、どんなことができるのか、いつもと同じように画面イメージによって説明しましょう。

最初、下に示すような**印刷メニュー画面**が現れます。登録してある全データ、男性・女性向け、コード入力による指定印刷、そして今回は職種別、年齢別による分類も加えてみました。

ダイレクトメールなどを発送する場合、発送先は一律ではなく、ある程度仕分けをしたい場合も出てくると思います。

印刷メニュー

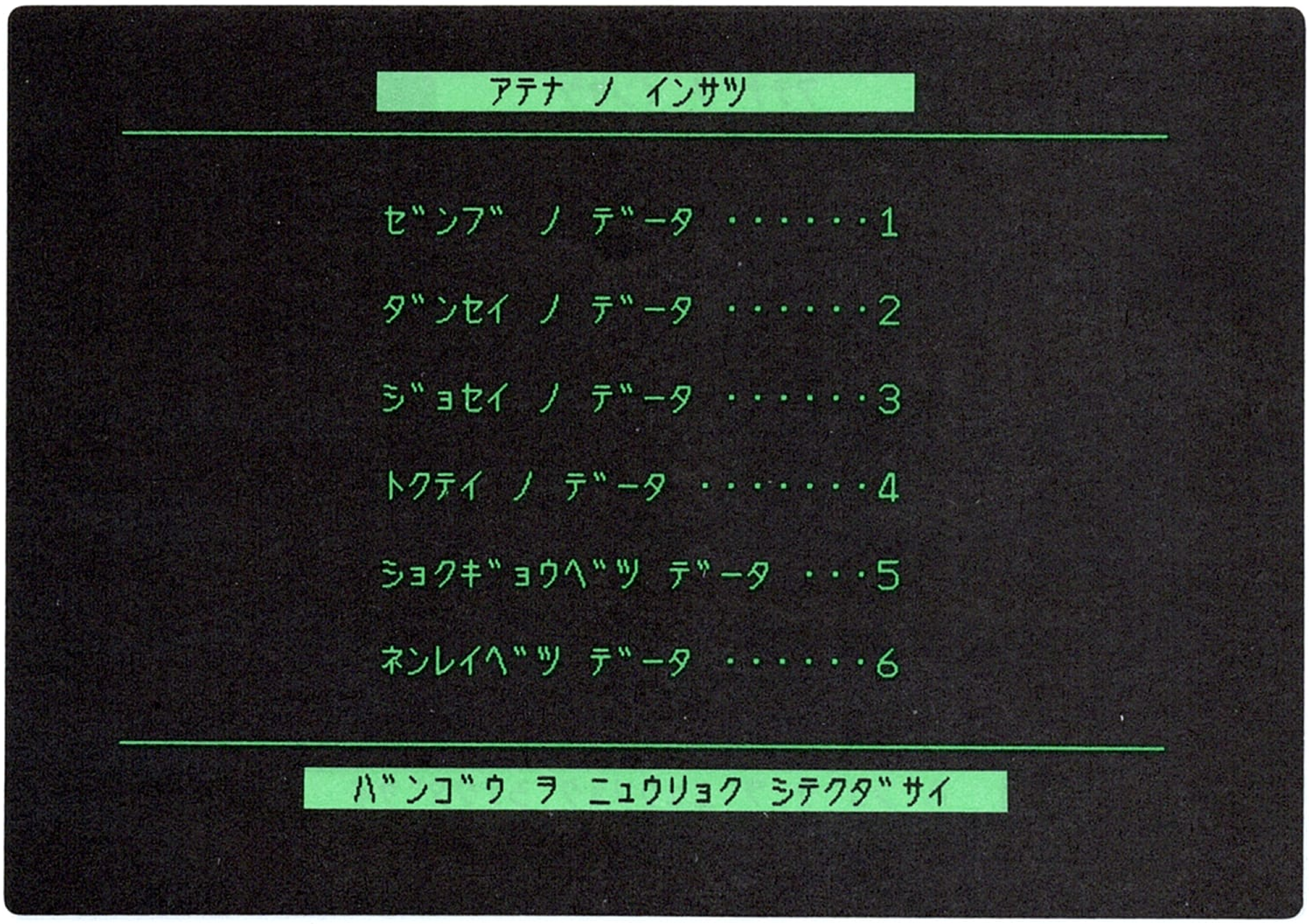


図１●印刷メニュー画面

メニューを選択すると画面は自動的に**図2**のように出力デバイスのメニューに変わります。

プリンタ、**CRTディスプレイ**、どちらでも出力できるようにしました。みなさんが本節でのプログラムを入力し、テストするときには印刷用紙もムダですから、まずディスプレイ表示を選択し、完全に動くようになってからプリンタへの出力を試されるとよいと思います。

全部のデータ、**男性のデータ**、**女性のデータ**を選んだ場合は、出力デバイスを指定するとただちに出力処理が行われます。プリンタへの出力は連続的に行われますが、ディスプレイへの出力は1件ずつリターンキーを押しながら進めていくようにしました。

印刷メニュー5の**職業別**を選択すると、デバイス指定ののち、**図3**のようなメニュー画面が現れます。メニューの種類は皆さんがあとでふやしてみてください。

印刷メニュー6の**年齢別**を選んだ場合は**図4**のような2種類のメニューが表示されます。

これもあとで調整してみてください。

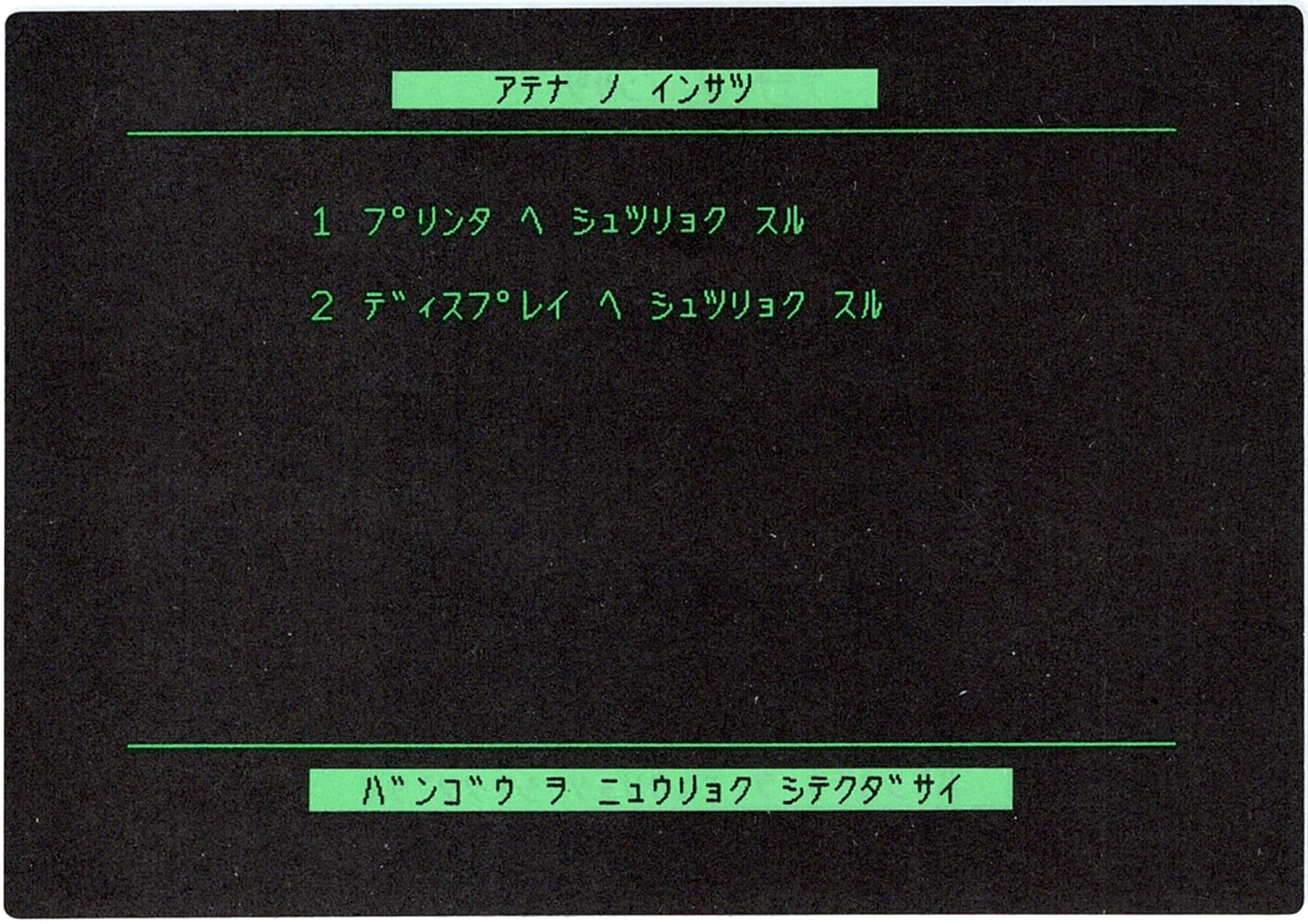


図2●出力デバイスの指定

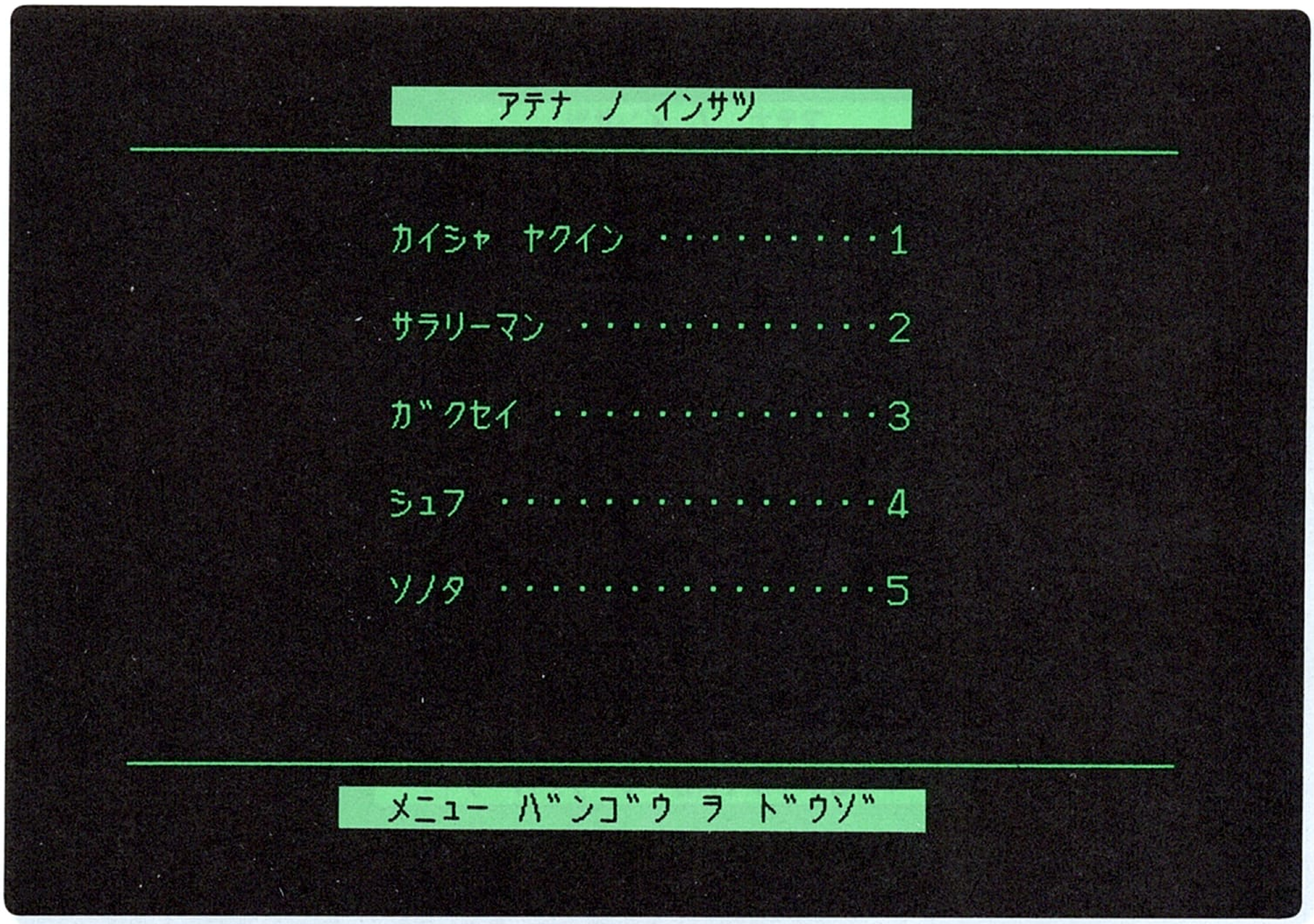


図 3 ●職業別印刷メニュー

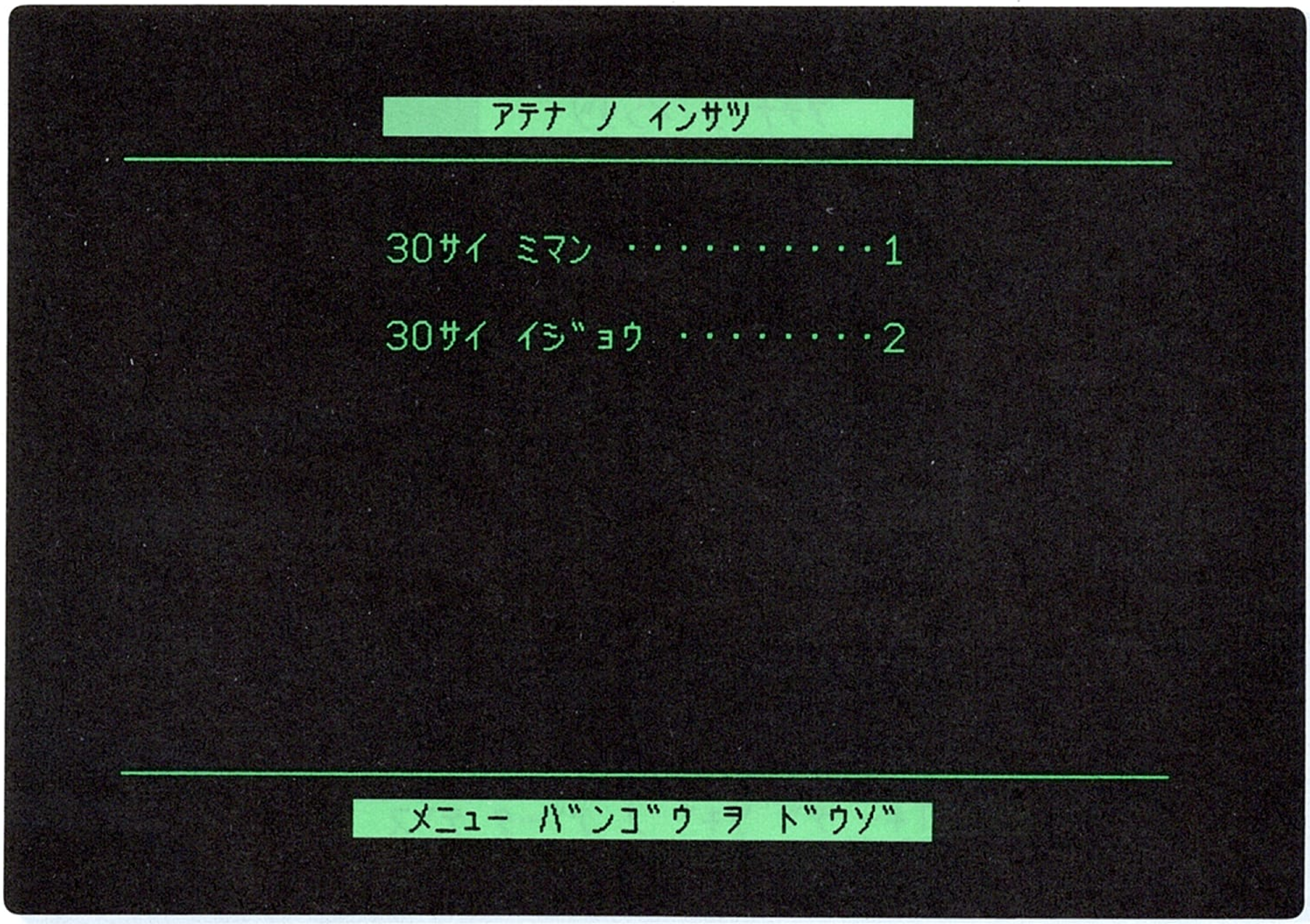


図 4 ●年齢別印刷メニュー

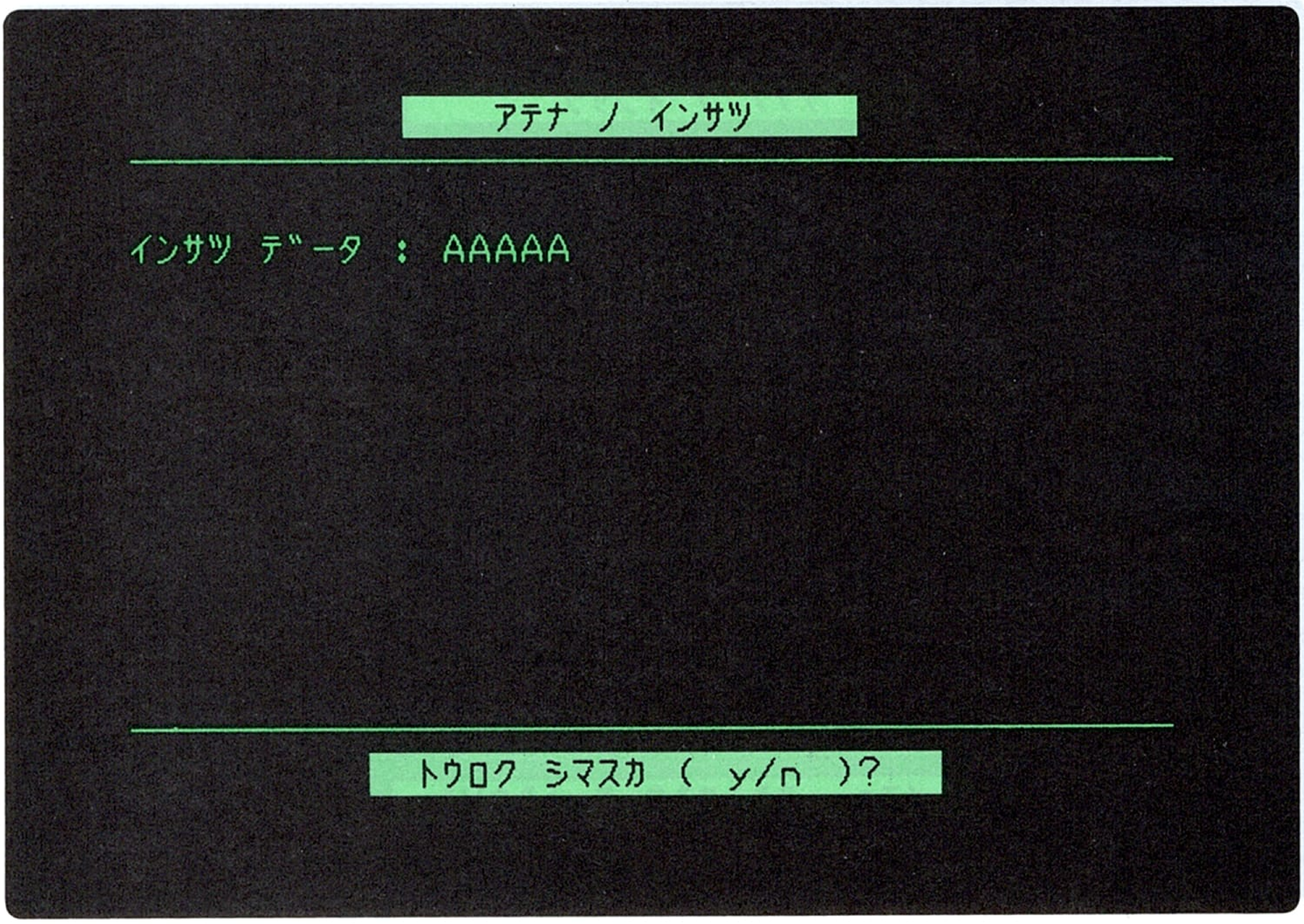


図5 ●印刷データの確認

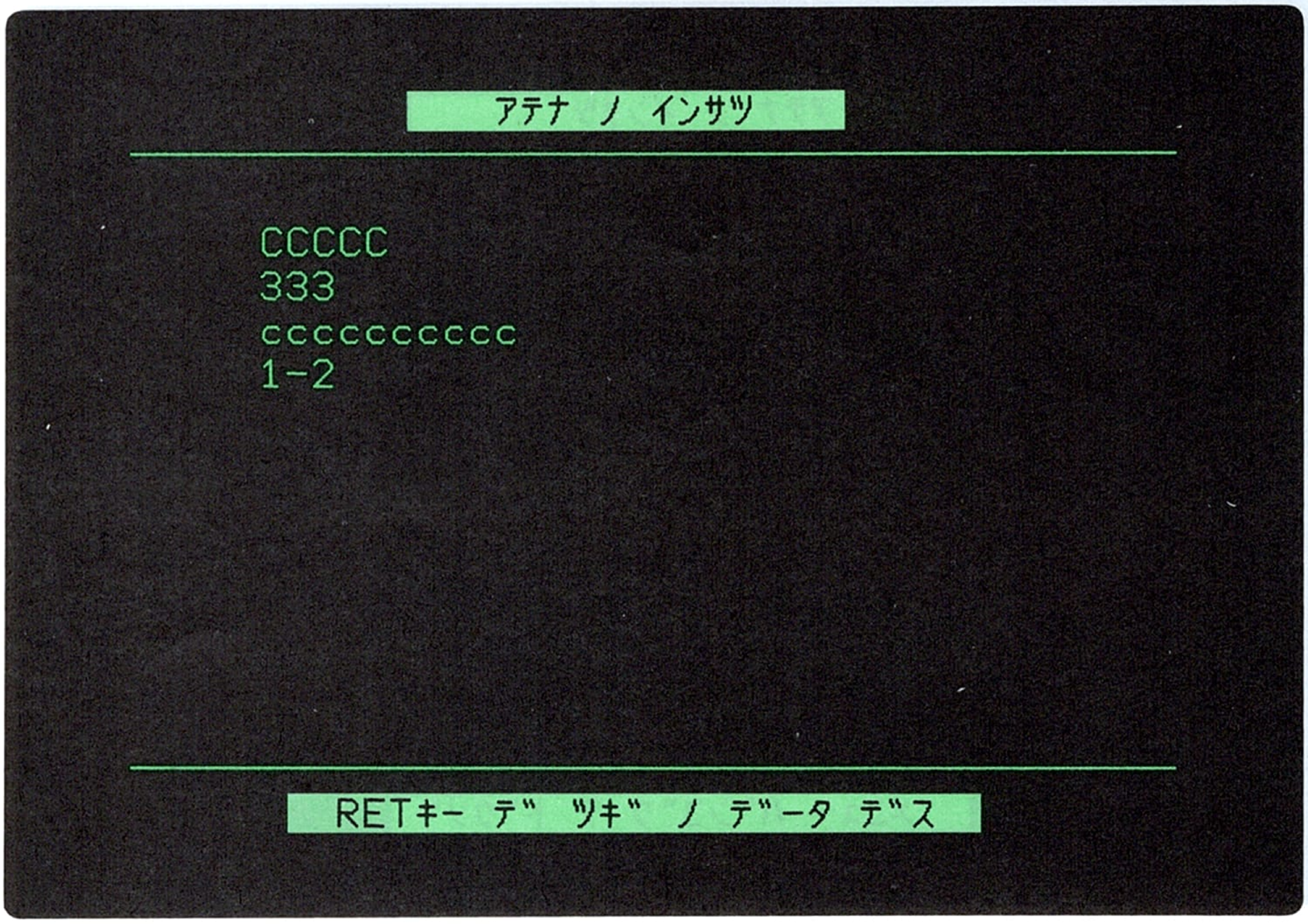


図6 ●CRTディスプレイへの出力

メニュー番号4の場合にはコード入力になります。コードを入力して、もし見つからなければエラーメッセージが表示されますし、該当コードがあれば**図5**のように名前が表示され、登録するかどうかの確認をうながします。

いつものように、コードに「**9999**」がキーインされるとプリンタまたはディスプレイへの出力が開始されます。

図6は**CRTディスプレイへの出力例**です。

また、次に示すのはメニュー番号2の男性のデータを選んで**プリンタへ印字**したものです。

```
111
aaaaaaaaaa
            1-1
     AAAAA
                 サマ

333
cccccccccc
            1-2
     CCCCC
                 サマ

555
eeeeeeeeee
            1-3
     EEEEE
                 サマ
```

初期設定

いつものように入口の部分から始めましょう。

```
100 '  file name ･･･> DMAIL4.BAS
110 '  1985.2.15
120 '  ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ
130 '  save "DMAIL4.BAS"
140 '
150 '————————————————————————————
160 *SHOKISETTEI
170 '————————————————————————————
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DEF FNSEC1(X)=(X-1) ¥ 50+2
210 DEF FNREC1(X)=(X-1) MOD 50+1
220 DEF FNSEC2(X)=(X-1) ¥ 3+12
230 DEF FNREC2(X)=(X-1) MOD 3+1
240 DIM NM$(3),PN$(3),AD$(3),BA$(3)
250 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
260 '
```

ファイルオープンとコード読み込み

このプログラムでは、出力すべきデータの登録順位を一時配列に格納し、あとでまとめて出力しますので、配列を宣言する必要がありますが、この大きさは、ランダムファイルの第1セクタにデータの総数

```
270 '————————————————————————————
280 *JYUNBI
290 '————————————————————————————
300 GOSUB *FOPEN3 ←———————— ランダムファイルオープン
310 GET #1,1
320 TOTAL=CVI(Q1$) ←——————— データ総数をメモリに取り出す
330 CLOSE #1
340 DIM B(TOTAL),CD(TOTAL) ←—— データ総数を使って配列宣言
350 GOSUB *YOMU ←—————————— 配列内にコードを読み込む
360 GOSUB *FOPEN1
370 GOSUB *FOPEN2
```

が書き込まれているので、これを使います。

配列Bは、出力するデータのディスク上での登録順位を一時格納しておくために用意しました。次は**ファイルオープンのルーチン**です。

```
960 '―――――――――――――――――――――――――――――――――
970 *FOPEN1
980 '―――――――――――――――――――――――――――――――――
990 OPEN "2:mail02.dat" AS #1
1000 FOR I=1 TO 50
1010   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
         2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
1020 NEXT I
1030 RETURN
1040 '
1050 '―――――――――――――――――――――――――――――――――
1060 *FOPEN2
1070 '―――――――――――――――――――――――――――――――――
1080 OPEN "2:mail02.dat" AS #2
1090 FOR I=1 TO 3
1100   FIELD #2,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1110 NEXT I
1120 RETURN
1130 '
1140 '―――――――――――――――――――――――――――――――――
1150 *FOPEN3
1160 '―――――――――――――――――――――――――――――――――
1170 OPEN "2:mail02.dat" AS #1
1180 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
1190 RETURN
1200 '
1210 '―――――――――――――――――――――――――――――――――
1220 *YOMU
1230 '―――――――――――――――――――――――――――――――――
1240 OPEN "2:indx02.dat" FOR INPUT AS #3
1250 WHILE NOT EOF(3)
1260   COUNT=COUNT+1
1270   INPUT #3,CD(COUNT)
1280 WEND
1290 CLOSE #3
1300 RETURN
1310 '
```

いつも繰り返していいますが、オープン文のファイル番号とフィールド文のバッファ番号を間違えないでください。

また、せっかくファイルのオープンおよびフィールド定義は正しくできていても、プログラム中でGETするときバッファ番号を間違えたりすると正しい値が得られません。

入出力画面とメニュー画面を作る

次に、私たちがコードを入力したり、あるいは結果を出力したりする**画面のレイアウトをする**ことにしましょう。

```
380 LOCATE 30,0 : COLOR 4
390 PRINT "     ｱﾃﾅ ﾉ ｲﾝｻﾂ       " : COLOR 0
400 LOCATE 20,1 : PRINT STRING$(40,149)
410 LOCATE 20,15: PRINT STRING$(40,149)
```

メニュー画面はサブルーチン処理にします。

```
420 GOSUB *MENU
```

```
1320 '―――――――――――――――――――――――――――――――――――
1330 *MENU
1340 '―――――――――――――――――――――――――――――――――――
1350 LOCATE 30,3 : PRINT "ｾﾞﾝﾌﾞ ﾉ ﾃﾞｰﾀ ･････1"
1360 LOCATE 30,5 : PRINT "ﾀﾞﾝｾｲ ﾉ ﾃﾞｰﾀ ･････2"
1370 LOCATE 30,7 : PRINT "ｼﾞｮｾｲ ﾉ ﾃﾞｰﾀ ･････3"
1380 LOCATE 30,9 : PRINT "ﾄｸﾃｲ ﾉ ﾃﾞｰﾀ ･････4"
1390 LOCATE 30,11: PRINT "ｼｮｸｷﾞｮｳﾍﾞﾂ ﾃﾞｰﾀ ･･5"
1400 LOCATE 30,13: PRINT "ﾈﾝﾚｲﾍﾞﾂ ﾃﾞｰﾀ ･････6"
1410 LOCATE 28,16: COLOR 6
1420 GOSUB *BUZZER
1430 PRINT " ﾊﾞﾝｺﾞｳ ｦ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ " : COLOR 0
1440 M$=INKEY$ : IF M$="" THEN GOTO 1440
1450 IF M$<"1" OR M$>"6" THEN BEEP : GOTO 1440
1460 GOSUB *KESU1
1470 RETURN
1480 '
```

163ページの画面表示を見ていただけばわかると思います。1450行は、

キー入力ミスに対するフォローです。

メニュー番号が入力されると、ただちに画面の一部を消去してジャンプ先へ戻ります。

画面消去、ブザー音のサブルーチン

いまのメニュー表示のルーチンでも出てきましたが、**画面の部分消去のサブルーチン**、それについでですから**ブザー音のサブルーチン**も入力しておきましょう。

```
2990 '――――――――――――――――――――
3000 *KESU1
3010 '――――――――――――――――――――
3020 FOR I=3 TO 13
3030   LOCATE 20,I : PRINT SPC(40)
3040 NEXT I
3050 RETURN
3060 '
3070 '――――――――――――――――――――
3080 *KESU2
3090 '――――――――――――――――――――
3100 LOCATE 20,16 : PRINT SPC(40)
3110 RETURN
3120 '
3130 '――――――――――――――――――――
3140 *BUZZER
3150 '――――――――――――――――――――
3160 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
3170 RETURN
```

＊**KESU 1** は上下の罫線の間、＊**KESU 2** は一番下のオペレーション表示行にスペースを出力して消去するサブルーチンです。

出力デバイスは？

次に出力するデバイスは何であるか確認をしておきます。

```
430 GOSUB *KAKUNIN
440 '
```

右ページにサブルーチンを示します。

```
1490 '────────────────────────────────
1500 *KAKUNIN
1510 '────────────────────────────────
1520 GOSUB *BUZZER ←──────────── ブザー音
1530 LOCATE 27,3
1540 PRINT "1 ﾌﾟﾘﾝﾀ ﾍ ｼｭﾂﾘｮｸ ｽﾙ"
1550 LOCATE 27,5
1560 PRINT "2 ﾃﾞｨｽﾌﾟﾚｲ ﾍ ｼｭﾂﾘｮｸ ｽﾙ"
1570 I$=INKEY$ : IF I$="" THEN GOTO 1570
1580 IF I$<"1" OR I$>"2" THEN BEEP : GOTO 1570
1590 IF I$="1" THEN INSATSU=-1 ←──── フラグを立てる
1600 GOSUB *KESU1 : GOSUB *KESU2 ←──┐
1610 RETURN                         │
1620 '              画面の部分消去 ──┘
```

もし1番を選択すると、1590行で行っているように**変数INSATSUに－1を代入し、値を「真」にしておきます。**あとで、ディスプレイ出力ルーチンへ飛ばすのか、プリンタ出力ルーチンへ飛ばすのか判定するためです。

メニュー番号に応じて分岐する

メニュー画面で番号を選択し、次いでデバイスを選択すると、それぞれのメニューに応じてサブルーチンへ分岐させます。

```
450 '────────────────────────────────
460 *BUNKI
470 '────────────────────────────────
480 ON INSTR("123456",M$) GOSUB *MENU1,*MENU2,
                   *MENU3,*MENU4,*MENU5,*MENU6
490 IF INSATSU THEN GOTO *PRT
500 GOTO *CRT
510 '
```

INSTR関数

480行に**INSTR(インストリング)関数**がありますが、文字列の中から任意の文字列を探して、その位置を与えるものです。整数配列の場合にはSEARCH関数がありましたが、これに対応するものです。この例の場合、170ページの1440行でメニュー番号入力はM＄で受けていますから、M＄の中には1～6のいずれかの数字（文字型）が格納されています。

ですから、もし2番を選択したのであれば

M$= "2"

ですから、これは、文字列

"123456"

の中の2番目にありますから、関数

INSTR ("123456",M$)

は「2」をその値として返してきます。つまりメニュー番号に応じて関数は

1, 2, 3, 4, 5, 6

を返してくれるわけです。

ところで皆さんは、

ON A GOSUB S1,S2,S3,…

という命令をごぞんじですね。Aに1から始まる整数が与えられたとき、1ならS1を、2ならS2を、3ならS3をというようにサブルーチンへジャンプする命令です。

これでおわかりと思いますが、私たちの例でも、メニュー番号によって、次のようにサブルーチンへジャンプするのです。

1……全部のデータ ──→ **MENU1**
2……男性のデータ ──→ **MENU2**
3……女性のデータ ──→ **MENU3**
4……特定のデータ ──→ **MENU4**
5……職業別 ──→ **MENU5**
6……年齢別 ──→ **MENU6**

そしてサブルーチンから戻ったら、出力ルーチンへ処理を移すようにします。

それではこれから、6つのサブルーチンを作っていくことにしましょう。

全データ印刷のサブルーチン

この部分は簡単です。登録データ数を320行でランダムファイルから取り出してありましたから、これをNUMBERに代入して戻せばよいのです。

```
1630 '————————————————————————————
1640 *MENU1
1650 '————————————————————————————
1660 NUMBER=TOTAL
1670 RETURN
1680 '
```

なぜわざわざ変数NUMBERに代入したかといいますと、6つのサブルーチンのいずれから帰ったときも印刷ルーチンは共通して使いたいからです。

つまり、出力はFOR～NEXT文で登録データ数だけ繰り返しますが、その最終値を変数NUMBERとしておくのです。

FOR I=1 TO NUMBER

ですから、どのサブルーチンでも帰るときに、出力データ数をこの変数に代入してリターンするようにします。

男性・女性のサブルーチン

```
1690 '————————————————————————————
1700 *MENU2
1710 '————————————————————————————
1720 OBJ1$="m" : OBJ2$="M"  ┐
1730 GOSUB *CHUUSHITSU      ├──男性の場合
1740 RETURN                 ┘
1750 '
1760 '————————————————————————————
1770 *MENU3
1780 '————————————————————————————
1790 OBJ1$="f" : OBJ2$="F"  ┐
1800 GOSUB *CHUUSHITSU      ├──女性の場合
1810 RETURN                 ┘
1820 '
```

この場合にはまず、データを抽出するためのフラグを変数に代入します。1720行、1790行がその処理です。そして、さらにサブルーチンにジャンプさせて印刷データを選び出すわけですが、このようにしておくと、**男性、女性の場合とも同じデータ抽出サブルーチンが使えます。**

次に、そのサブルーチンを示します。「抽出」という表現が当たっているかどうかわかりませんが、とにかく**ディスクから男性または女性のデータを選び出す**のです。

```
1830 '─────────────────────────
1840 *CHUUSHITSU
1850 '─────────────────────────
1860 S.LENGTH=(TOTAL-1) ¥ 50+2 ←── 使用セクタ数の算出
1870 FOR I=2 TO S.LENGTH
1880   GET #1,I
1890   FOR J=1 TO 50
1900     IF SEX$(J)=OBJ1$ OR SEX$(J)=OBJ2$        ┐
           THEN  TR=TR+1 : B(TR)=(I-2)*50+J       ┘
                                    一致したフラグがあれば登録
1910   NEXT J
1920 NEXT I
1930 NUMBER=TR ←── 登録数を代入
1940 RETURN
1950 '
```

まず、1860行で全データ数をもとにして、性別などが書き込んであるセクタをどこまで使用しているか、その**最終セクタを算出**します。そして、第2セクタから最終セクタまでをループさせる大きなFOR～NEXTを作り、その内側に1ステップずつ50回（1セクタに50レコード書き込めました）ループする小さなFOR～NEXTを作り、各レコードの性別用のフラグをチェックしていきます。

〈注〉ランダムファイルの使用セクタ数はLOF関数でもわかりますが、この場合全体のファイル長がわかるのであって、私たちの場合のように、第1セクタと第12セクタの間にはさまれたときは判明しません。

もし該当するフラグが見つかればカウンタTRを1つインクリメント（加算）し、そのデータの**登録順位**（ディスク上での）を配列Bにしまいます。

登録順位

ここでいう登録順位とは、122ページの表でたとえれば、「EEEEE」さんなら「5」、「GGGGG」さんなら「7」ということです。

この登録順位の算出方法ですが、まずそのデータが書き込まれているセクタの前に、同じフォーマットのセクタがいくつあるか調べます。それは、第2セクタから始まりますから、

$$\overset{アイ}{I}-2$$

によって算出できます。
つまり、あるデータが第2セクタに書き込まれているのでしたら

2−2=0

となりますから、1つもありません。
第4セクタに書き込まれているデータでしたら

4−2

ですから、その前に**2つのセクタ**があります。つまり第2、第3セクタです。したがって、この2つのセクタには**レコード**が

(4−2)×50

だけ書き込まれています。そして、第4セクタの**任意のレコード**は、ループ変数Jを使って、

(4−2)×50+J

と表わせます。
いまは第4セクタを例にとって説明しましたが、任意のセクタはループ変数Iで代用できますから、上の式の4をIに置き換えて、

(I−2)×50+J

という式を得ることができます。パソコンでは乗算記号はアスタリスクを使いますから、

(I−2)*50+J

になります。
この値を配列に代入すれば、配列の各要素に出力すべきデータのディスク上での登録順位がおさめられていきます。
たとえば122ページの表で女性の場合でしたら2、4、6番目の3つのデータがありますから、配列Bには、

B(1)←── 2
B(2)←── 4
B(3)←── 6

というように格納されます。そして、この登録数3がTRに入っていますから、これを変数NUMBERに代入しなおして、このサブルーチンから帰ります。

コード入力によるサブルーチン

```
1960 '―――――――――――――――――――――――――――
1970 *MENU4
1980 '―――――――――――――――――――――――――――
1990 *LOOP
2000 GOSUB *KESU1 ←――――――――入力行のクリア
2010 LOCATE 40,3 : COLOR 6
2020 PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  " : COLOR 0
2030 LOCATE 20,3 : INPUT "ｺｰﾄﾞ";CD
2040 IF CD=9999 THEN GOTO 2100
2050 GOSUB *SAGASU ←――――――――コードのサーチ
2060 IF P=-1 THEN GOTO *LOOP ←――――――見つからない
2070 IF NOT TOOROKU THEN GOTO *LOOP ←―――登録しない
2080 TR=TR+1 : B(TR)=P ←――――ディスク上での登録順を配列に
2090 TOOROKU=0 : GOTO *LOOP
2100 NUMBER=TR
2110 GOSUB *KESU1 : GOSUB *KESU2
2120 RETURN
2130 '
2140 '―――――――――――――――――――――――――――
2150 *SAGASU
2160 '―――――――――――――――――――――――――――
2170 P=SEARCH(CD,CD) ←―――――――配列内にコードがあるか？
2180 IF P=-1 THEN GOSUB *MITOOROKU ←――見つからない
2190 SEC2=FNSEC2(P) : REC2=FNREC2(P) ┐
2200 GET #2,SEC2                     ┘
2210 GOSUB *BUZZER 見つかったのでセクタ番号、セクタ内での
2220 GOSUB *KESU1  レコード番号を算出してセクタをGET
2230 LOCATE 20,3 : PRINT "ｲﾝｻﾂ ﾃﾞｰﾀ : "; ┐
2240 COLOR 4 : PRINT NM$(REC2) : COLOR 0 │
2250 LOCATE 30,16 : COLOR 6              ├―確認
2260 PRINT " ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?  "       ┘
2270 COLOR 0
2280 I$=INKEY$ : IF I$="" THEN GOTO 2280
2290 IF I$="Y" OR I$="y" THEN GOTO 2320
2300 IF I$="N" OR I$="n" THEN GOTO 2330
2310 BEEP : BEEP : GOTO 2280
2320 TOOROKU=-1 ←―――――――――――フラグ「真」にする
2330 GOSUB *KESU1 : GOSUB *KESU2
2340 RETURN
2350 '
```

プログラムの意味は、コメントを入れておきましたので理解できると思います。コードを入力して、それをサーチする場合の決まり文句といったところです。

次に**該当コードがなかった場合のサブルーチン**を示します。2180行からジャンプしてきます。

```
2360 '────────────────────────
2370 *MITOOROKU
2380 '────────────────────────
2390 BEEP : BEEP : BEEP
2400 GOSUB *KESU1
2410 LOCATE 30,3 : COLOR 6
2420 PRINT "   ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ !  " : COLOR 0
2430 FOR I=1 TO 2000 : NEXT I
2440 GOSUB *KESU1
2450 RETURN 2340
2460 '
```

職種別のサブルーチン

このサブルーチンも、男女別のサブルーチンと理屈はまったく同じです。

職種を選択しなければいけませんので、メニュー画面を加えておきましょう。

右ページにプログラムリストを示しますのでキーインしてください。

各職種に対応する番号は、ランダムファイルへ書き込むとき、数値データとしてではなく、ストリングとして扱いましたので、2590行の**INKEY$関数**で受けています。

2600行のステートメントは指定外の番号が入力された場合の処理で、ビープ音を鳴らして2590行に戻しています。

このサブルーチンでは、2510行と2690行でメニュー表示行の消去とオペレーション表示行の消去を行っています。余計な部分もあるかもしれませんが、たいして処理時間もかかりませんので入れておきました。

```
2470 '―――――――――――――――――――――――――――――――
2480 *MENU5
2490 '―――――――――――――――――――――――――――――――
2500 GOSUB *BUZZER
2510 GOSUB *KESU1 : GOSUB *KESU2
2520 LOCATE 27,16 : COLOR 6
2530 PRINT "   ﾒﾆｭｰ ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ   " : COLOR 0
2540 LOCATE 30,3 : PRINT "ｶｲｼｬ ﾔｸｲﾝ ・・・・・・・・1"
2550 LOCATE 30,5 : PRINT "ｻﾗﾘｰﾏﾝ ・・・・・・・・・・2"
2560 LOCATE 30,7 : PRINT "ｶﾞｸｾｲ ・・・・・・・・・・・3"
2570 LOCATE 30,9 : PRINT "ｼｭﾌ ・・・・・・・・・・・・4"
2580 LOCATE 30,11: PRINT "ｿﾉﾀ ・・・・・・・・・・・・5"
2590 PF$=INKEY$ : IF PF$="" THEN GOTO 2590
2600 IF PF$<"1" OR PF$>"5" THEN BEEP : GOTO 2590
2610 S.LENGTH=(TOTAL-1) ¥ 50+2  ←使用セクタ数の算出
2620 FOR I=2 TO S.LENGTH
2630   GET #1,I
2640   FOR J=1 TO 50
2650     IF PRF$(J)=PF$ THEN TR=TR+1          ┐
                     : B(TR)=(I-2)*50+J       ┘←見つかれば登録
2660   NEXT J
2670 NEXT I
2680 NUMBER=TR
2690 GOSUB*KESU1 : GOSUB *KESU2
2700 RETURN
2710 '
```

年齢区分によるサブルーチン

やはりこの場合も、いまの職種別サブルーチンとメニュー画面が異なるくらいで、そのほかはほとんど同じです。

プログラムを次ページに示します。

メニューは2種類に限定しました。30歳未満と30歳以上です。

もう少し分類したいという方は、本章が終ってからチャレンジしてみてください。

2900行、2910行が年齢判定の条件文ですが、ここで注意しなくてはいけないのは30歳未満の場合です。

```
2720 '────────────────────────────────────
2730 *MENU6
2740 '────────────────────────────────────
2750 GOSUB *KESU1 : GOSUB *KESU2
2760 LOCATE 29,16 : COLOR 6
2770 PRINT " ﾒﾆｭｰ ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ  " : COLOR 0
2780 LOCATE 30,3 : PRINT "30ｻｲ ﾐﾏﾝ ..........1"
2790 LOCATE 30,5 : PRINT "30ｻｲ ｲｼﾞｮｳ ........2"
2800 GOSUB *BUZZER
2810 AG$=INKEY$ : IF AG$="" THEN GOTO 2810
2820 IF AG$<"1" OR AG$>"2" THEN BEEP : GOTO 2810
2830 S.LENGTH=(TOTAL-1) ¥ 50+2
2840 FOR I=2 TO S.LENGTH
2850   GET #1,I
2860   FOR J=1 TO 50
2870     C=C+1 ←─────────────カウンタ
2880     IF C>TOTAL THEN GOTO 2940 ←─登録総数を超えたら終り
2890     IF AG$="2" THEN GOTO 2910
2900     IF CVI(AGE$(J))>0 AND CVI(AGE$(J))<30
           THEN GOTO 2920 ELSE GOTO 2930
2910     IF CVI(AGE$(J))>=30 THEN GOTO 2920
                             ELSE GOTO 2930
2920     TR=TR+1 : B(TR)=(I-2)*50+J ←────登録
2930   NEXT J
2940 NEXT I
2950 NUMBER=TR
2960 GOSUB *KESU1 : GOSUB *KESU2
2970 RETURN
2980 '
```

30歳未満ですから、

CVI(AGE$(J))<30

という条件だけでよさそうに見えますが、「**0より大きく**」という条件もつけておきませんと、おかしなことになってしまいます。

なぜかと申しますと、ランダムファイルへの書き込み時にデータをLSETしてPUTしますと、そのセクタの残りの部分には0が書き込まれます。たとえば、私たちの場合ですと、第2セクタには7つのデータしか書き込んでありませんから、残りの43のデータを書き込む箇所には0が入ります。

そうしますと、もし**「0より大きく」**(0は含まれません)という条件がないと、8番目のデータから50番目のデータまですべて0ですから、どれも2900行の条件を満たすことになり、このプログラムですと2920行で、

Subscript out of range in 2920

のエラーとなると思います。

つまり、**どのデータも登録されてしまうことになり、配列が不足してしまう**わけです。

このプログラムでは配列を最小限にとってありましたからまだいいのですが、たとえば配列を50くらいとってありますと、このサブルーチンではエラーにならず、印刷ルーチンへ行って、おかしな出力をすることになります。

こんなときのバグはちょっと探しにくいですよ。

プリンタへの出力ルーチン

以上で、入力すべきサブルーチンはすべて終りました。残すは出力ルーチンです。まずプリンタへの出力ルーチンから……。

```
520 '────────────────────
530 *PRT
540 '────────────────────
550 FOR I=1 TO NUMBER
560   IF B(I)=0 THEN P=I : GOTO 580
570   P=B(I)
580   SEC2=FNSEC2(P)
590   REC2=FNREC2(P)
600   GET #2,SEC2
610   LPRINT PN$(REC2)
620   LPRINT AD$(REC2)
630   LPRINT TAB(10);BA$(REC2)
640   LPRINT TAB(5);NM$(REC2)
650   LPRINT TAB(15);"サマ"
660   FOR J=1 TO 5 : LPRINT : NEXT J
670 NEXT I
680 GOTO *OWARI
690 '
```

第1章でも勉強しましたからわかりますね。

変数NUMBERに出力データ数が入っていますから、この数だけループさせて出力します。

ディスプレイへの出力と終りの処理

次に**CRTディスプレイへの出力**です。がんばって入力してしまいましょう。

```
700 '――――――――――――――――――――――――――――――
710 *CRT
720 '――――――――――――――――――――――――――――――
730 COLOR 6 : LOCATE 27,16
740 PRINT " RETｷｰ ﾃﾞ ﾂｷﾞ ﾉ ﾃﾞｰﾀ ﾃﾞｽ "
750 COLOR 0
760 FOR I=1 TO NUMBER
770   IF B(I)=0 THEN P=I : GOTO 790
780   P=B(I)
790   SEC2=FNSEC2(P)
800   REC2=FNREC2(P)
810   GET #2,SEC2
820   LOCATE 25,3 : PRINT NM$(REC2)
830   LOCATE 25,4 : PRINT PN$(REC2)
840   LOCATE 25,5 : PRINT AD$(REC2)
850   LOCATE 25,6 : PRINT BA$(REC2)
860   I$=INKEY$ : IF I$="" THEN GOTO 860
870 NEXT I
880 '
890 '――――――――――――――――――――――――――――――
900 *OWARI
910 '――――――――――――――――――――――――――――――
920 CLOSE
930 CLS
940 RUN "menu2.bas"
950 '
```

それでは**プログラムをセーブ**してから、163〜166ページの画面を見ながら実行してみてください。次ページにプログラム総リストを示します。

```
save "DMAIL4.BAS" ⏎
```

```
100 ' file name ····> DMAIL4.BAS
110 ' 1985.2.15
120 ' ﾗﾝﾀﾞﾑ ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ
130 ' save "DMAIL4.BAS"
140 '
150 '────────────────────────────────────
160 *SHOKISETTEI
170 '────────────────────────────────────
180 WIDTH 80,25 : CONSOLE 0,25,0,0
190 DEFINT A-Z
200 DEF FNSEC1(X)=(X-1) ¥ 50+2
210 DEF FNREC1(X)=(X-1) MOD 50+1
220 DEF FNSEC2(X)=(X-1) ¥ 3+12
230 DEF FNREC2(X)=(X-1) MOD 3+1
240 DIM NM$(3),PN$(3),AD$(3),BA$(3)
250 DIM SEX$(50),AGE$(50),PRF$(50),SPR$(50)
260 '
270 '────────────────────────────────────
280 *JYUNBI
290 '────────────────────────────────────
300 GOSUB *FOPEN3
310 GET #1,1
320 TOTAL=CVI(Q1$)
330 CLOSE #1
340 DIM B(TOTAL),CD(TOTAL)
350 GOSUB *YOMU
360 GOSUB *FOPEN1
370 GOSUB *FOPEN2
380 LOCATE 30,0 : COLOR 4
390 PRINT "    ｱﾃﾅ ﾉ ｲﾝｻﾂ      " : COLOR 0
400 LOCATE 20,1 : PRINT STRING$(40,149)
410 LOCATE 20,15: PRINT STRING$(40,149)
420 GOSUB *MENU
430 GOSUB *KAKUNIN
440 '
450 '────────────────────────────────────
460 *BUNKI
470 '────────────────────────────────────
480 ON INSTR("123456",M$) GOSUB *MENU1,*MENU2,
              *MENU3,*MENU4,*MENU5,*MENU6
490 IF INSATSU THEN GOTO *PRT
```

```
500 GOTO *CRT
510 '
520 '――――――――――――――――――――――――――――――――――――――――
530 *PRT
540 '――――――――――――――――――――――――――――――――――――――――
550 FOR I=1 TO NUMBER
560   IF B(I)=0 THEN P=I : GOTO 580
570   P=B(I)
580   SEC2=FNSEC2(P)
590   REC2=FNREC2(P)
600   GET #2,SEC2
610   LPRINT PN$(REC2)
620   LPRINT AD$(REC2)
630   LPRINT TAB(10);BA$(REC2)
640   LPRINT TAB(5);NM$(REC2)
650   LPRINT TAB(15);"ｻﾏ"
660   FOR J=1 TO 5 : LPRINT : NEXT J
670 NEXT I
680 GOTO *OWARI
690 '
700 '――――――――――――――――――――――――――――――――――――――――
710 *CRT
720 '――――――――――――――――――――――――――――――――――――――――
730 COLOR 6 : LOCATE 27,16
740 PRINT " RETｷｰ ﾃﾞ ﾂｷﾞ ﾉ ﾃﾞｰﾀ ﾃﾞｽ  "
750 COLOR 0
760 FOR I=1 TO NUMBER
770   IF B(I)=0 THEN P=I : GOTO 790
780   P=B(I)
790   SEC2=FNSEC2(P)
800   REC2=FNREC2(P)
810   GET #2,SEC2
820   LOCATE 25,3 : PRINT NM$(REC2)
830   LOCATE 25,4 : PRINT PN$(REC2)
840   LOCATE 25,5 : PRINT AD$(REC2)
850   LOCATE 25,6 : PRINT BA$(REC2)
860   I$=INKEY$ : IF I$="" THEN GOTO 860
870 NEXT I
880 '
890 '――――――――――――――――――――――――――――――――――――――――
900 *OWARI
```

```
910 '————————————————————————————————
920 CLOSE
930 CLS
940 RUN "menu2.bas"
950 '
960 '————————————————————————————————
970 *FOPEN1
980 '————————————————————————————————
990 OPEN "2:mail02.dat" AS #1
1000 FOR I=1 TO 50
1010   FIELD #1,(I-1)*5 AS D$,1 AS SEX$(I),
         2 AS AGE$(I),1 AS PRF$(I),1 AS SPR$(I)
1020 NEXT I
1030 RETURN
1040 '
1050 '————————————————————————————————
1060 *FOPEN2
1070 '————————————————————————————————
1080 OPEN "2:mail02.dat" AS #2
1090 FOR I=1 TO 3
1100   FIELD #2,85*(I-1) AS D$,20 AS NM$(I),
          6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
1110 NEXT I
1120 RETURN
1130 '
1140 '————————————————————————————————
1150 *FOPEN3
1160 '————————————————————————————————
1170 OPEN "2:mail02.dat" AS #1
1180 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
             2 AS Q4$,2 AS Q5$,2 AS Q6$
1190 RETURN
1200 '
1210 '————————————————————————————————
1220 *YOMU
1230 '————————————————————————————————
1240 OPEN "2:indx02.dat" FOR INPUT AS #3
1250 WHILE NOT EOF(3)
1260   COUNT=COUNT+1
1270   INPUT #3,CD(COUNT)
1280 WEND
```

```
1290 CLOSE #3
1300 RETURN
1310 '
1320 '――――――――――――――――――――――――――――――――――――――――
1330 *MENU
1340 '――――――――――――――――――――――――――――――――――――――――
1350 LOCATE 30,3 : PRINT "ｾﾞﾝﾌﾞ ﾉ ﾃﾞｰﾀ ･･････1"
1360 LOCATE 30,5 : PRINT "ﾀﾞﾝｾｲ ﾉ ﾃﾞｰﾀ ･･････2"
1370 LOCATE 30,7 : PRINT "ｼﾞｮｾｲ ﾉ ﾃﾞｰﾀ ･･････3"
1380 LOCATE 30,9 : PRINT "ﾄｸﾃｲ ﾉ ﾃﾞｰﾀ ･･･････4"
1390 LOCATE 30,11: PRINT "ｼｮｸｷﾞｮｳﾍﾞﾂ ﾃﾞｰﾀ ･･･5"
1400 LOCATE 30,13: PRINT "ﾈﾝﾚｲﾍﾞﾂ ﾃﾞｰﾀ ･･････6"
1410 LOCATE 28,16: COLOR 6
1420 GOSUB *BUZZER
1430 PRINT " ﾊﾞﾝｺﾞｳ ｦ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ " : COLOR 0
1440 M$=INKEY$ : IF M$="" THEN GOTO 1440
1450 IF M$<"1" OR M$>"6" THEN BEEP : GOTO 1440
1460 GOSUB *KESU1
1470 RETURN
1480 '
1490 '――――――――――――――――――――――――――――――――――――――――
1500 *KAKUNIN
1510 '――――――――――――――――――――――――――――――――――――――――
1520 GOSUB *BUZZER
1530 LOCATE 27,3
1540 PRINT "1 ﾌﾟﾘﾝﾀ ﾍ ｼｭﾂﾘｮｸ ｽﾙ"
1550 LOCATE 27,5
1560 PRINT "2 ﾃﾞｨｽﾌﾟﾚｲ ﾍ ｼｭﾂﾘｮｸ ｽﾙ"
1570 I$=INKEY$ : IF I$="" THEN GOTO 1570
1580 IF I$<"1" OR I$>"2" THEN BEEP : GOTO 1570
1590 IF I$="1" THEN INSATSU=-1
1600 GOSUB *KESU1 : GOSUB *KESU2
1610 RETURN
1620 '
1630 '――――――――――――――――――――――――――――――――――――――――
1640 *MENU1
1650 '――――――――――――――――――――――――――――――――――――――――
1660 NUMBER=TOTAL
1670 RETURN
1680 '
1690 '――――――――――――――――――――――――――――――――――――――――
```

```
1700 *MENU2
1710 '--------------------------------------------------------
1720 OBJ1$="m"  : OBJ2$="M"
1730 GOSUB *CHUUSHITSU
1740 RETURN
1750 '
1760 '--------------------------------------------------------
1770 *MENU3
1780 '--------------------------------------------------------
1790 OBJ1$="f" : OBJ2$="F"
1800 GOSUB *CHUUSHITSU
1810 RETURN
1820 '
1830 '--------------------------------------------------------
1840 *CHUUSHITSU
1850 '--------------------------------------------------------
1860 S.LENGTH=(TOTAL-1) ¥ 50+2
1870 FOR I=2 TO S.LENGTH
1880   GET #1,I
1890   FOR J=1 TO 50
1900     IF SEX$(J)=OBJ1$ OR SEX$(J)=OBJ2$
           THEN  TR=TR+1 : B(TR)=(I-2)*50+J
1910   NEXT J
1920 NEXT I
1930 NUMBER=TR
1940 RETURN
1950 '
1960 '--------------------------------------------------------
1970 *MENU4
1980 '--------------------------------------------------------
1990 *LOOP
2000 GOSUB *KESU1
2010 LOCATE 40,3 : COLOR 6
2020 PRINT " ｵﾜﾘ ﾉ ﾄｷ ﾊ 9999  " : COLOR 0
2030 LOCATE 20,3 : INPUT "ｺｰﾄﾞ ";CD
2040 IF CD=9999 THEN GOTO 2100
2050 GOSUB *SAGASU
2060 IF P=-1 THEN GOTO *LOOP
2070 IF NOT TOOROKU THEN GOTO *LOOP
2080 TR=TR+1 : B(TR)=P
2090 TOOROKU=0 : GOTO *LOOP
```

```
2100 NUMBER=TR
2110 GOSUB *KESU1 : GOSUB *KESU2
2120 RETURN
2130 '
2140 '――――――――――――――――――――――――――――――
2150 *SAGASU
2160 '――――――――――――――――――――――――――――――
2170 P=SEARCH(CD,CD)
2180 IF P=-1 THEN GOSUB *MITOOROKU
2190 SEC2=FNSEC2(P) : REC2=FNREC2(P)
2200 GET #2,SEC2
2210 GOSUB *BUZZER
2220 GOSUB *KESU1
2230 LOCATE 20,3 : PRINT "ｲﾝｻﾂ ﾃﾞｰﾀ : ";
2240 COLOR 4 : PRINT NM$(REC2) : COLOR 0
2250 LOCATE 30,16 : COLOR 6
2260 PRINT " ﾄｳﾛｸ ｼﾏｽｶ ( y/n )?  "
2270 COLOR 0
2280 I$=INKEY$ : IF I$="" THEN GOTO 2280
2290 IF I$="Y" OR I$="y" THEN GOTO 2320
2300 IF I$="N" OR I$="n" THEN GOTO 2330
2310 BEEP : BEEP : GOTO 2280
2320 TOOROKU=-1
2330 GOSUB *KESU1 : GOSUB *KESU2
2340 RETURN
2350 '
2360 '――――――――――――――――――――――――――――――
2370 *MITOOROKU
2380 '――――――――――――――――――――――――――――――
2390 BEEP : BEEP : BEEP
2400 GOSUB *KESU1
2410 LOCATE 30,3 : COLOR 6
2420 PRINT "   ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ !  " : COLOR 0
2430 FOR I=1 TO 2000 : NEXT I
2440 GOSUB *KESU1
2450 RETURN 2340
2460 '
2470 '――――――――――――――――――――――――――――――
2480 *MENU5
2490 '――――――――――――――――――――――――――――――
2500 GOSUB *BUZZER
```

```
2510 GOSUB *KESU1 : GOSUB *KESU2
2520 LOCATE 27,16 : COLOR 6
2530 PRINT "    ﾒﾆｭｰ ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ       " : COLOR 0
2540 LOCATE 30,3 : PRINT "ｶｲｼｬ ﾔｸｲﾝ ··········1"
2550 LOCATE 30,5 : PRINT "ｻﾗﾘｰﾏﾝ ············2"
2560 LOCATE 30,7 : PRINT "ｶﾞｸｾｲ ·············3"
2570 LOCATE 30,9 : PRINT "ｼｭﾌ ···············4"
2580 LOCATE 30,11: PRINT "ｿﾉﾀ ···············5"
2590 PF$=INKEY$ : IF PF$="" THEN GOTO 2590
2600 IF PF$<"1" OR PF$>"5" THEN BEEP : GOTO 2590
2610 S.LENGTH=(TOTAL-1) ¥ 50+2
2620 FOR I=2 TO S.LENGTH
2630   GET #1,I
2640   FOR J=1 TO 50
2650     IF PRF$(J)=PF$ THEN TR=TR+1
                          : B(TR)=(I-2)*50+J
2660   NEXT J
2670 NEXT I
2680 NUMBER=TR
2690 GOSUB*KESU1 : GOSUB *KESU2
2700 RETURN
2710 '
2720 '───────────────────────────────────
2730 *MENU6
2740 '───────────────────────────────────
2750 GOSUB *KESU1 : GOSUB *KESU2
2760 LOCATE 29,16 : COLOR 6
2770 PRINT " ﾒﾆｭｰ ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ    " : COLOR 0
2780 LOCATE 30,3 : PRINT "30ｻｲ ﾐﾏﾝ ··········1"
2790 LOCATE 30,5 : PRINT "30ｻｲ ｲｼﾞｮｳ ········2"
2800 GOSUB *BUZZER
2810 AG$=INKEY$ : IF AG$="" THEN GOTO 2810
2820 IF AG$<"1" OR AG$>"2" THEN BEEP : GOTO 2810
2830 S.LENGTH=(TOTAL-1) ¥ 50+2
2840 FOR I=2 TO S.LENGTH
2850   GET #1,I
2860   FOR J=1 TO 50
2870     C=C+1
2880     IF C>TOTAL THEN GOTO 2940
2890     IF AG$="2" THEN GOTO 2910
2900     IF CVI(AGE$(J))>0 AND CVI(AGE$(J))<30
            THEN GOTO 2920 ELSE GOTO 2930
```

```
2910        IF CVI(AGE$(J))>=30 THEN GOTO 2920
                                  ELSE GOTO 2930
2920        TR=TR+1 : B(TR)=(I-2)*50+J
2930      NEXT J
2940    NEXT I
2950    NUMBER=TR
2960    GOSUB *KESU1 : GOSUB *KESU2
2970    RETURN
2980    '
2990    '————————————————————————————————————————
3000    *KESU1
3010    '————————————————————————————————————————
3020    FOR I=3 TO 13
3030      LOCATE 20,I : PRINT SPC(40)
3040    NEXT I
3050    RETURN
3060    '
3070    '————————————————————————————————————————
3080    *KESU2
3090    '————————————————————————————————————————
3100    LOCATE 20,16 : PRINT SPC(40)
3110    RETURN
3120    '
3130    '————————————————————————————————————————
3140    *BUZZER
3150    '————————————————————————————————————————
3160    FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
3170    RETURN
```

2・10　メニュープログラム

それではこの章も、**メニュープログラム**を作って終りです。第1章のメニューと同じようなものですので、まず前章の

menu1.bas

をロードし、スクリーンエディタで変更してください。**変更点**は、

❶　100行、130行のプログラム名
❷　640行のデータ数
❸　650〜690行のデータ文のメニュー番号および実行するファイル名
❹　665行をあらたに追加してください。

以上の修正をほどこしたのち、このプログラムをセーブしてください。リストを次に示します。

```
100 ' file name ････> menu2.bas
110 ' 1985.2.10
120 ' ｱﾃﾅ ｲﾝｻﾂ ﾒﾆｭｰ ﾌﾟﾛｸﾞﾗﾑ
130 ' save "menu2.bas"
140 '
150 WIDTH 80,20 : CONSOLE 0,20,0,0
160 DEFINT A-Z
170 '
180 READ MENU
190 DIM MENU$(MENU),PROG$(MENU)
200 FOR I=1 TO MENU
210   READ MENU$(I),PROG$(I)
220 NEXT I
230 '
240 PRINT TAB(18)
250 PRINT STRING$(40,149)
260 PRINT TAB(32)
270 PRINT"ｱﾃﾅ ｲﾝｻﾂ ﾒﾆｭｰ"
280 PRINT TAB(18)
290 PRINT STRING$(40,149)
300 PRINT
310 '
320 FOR I=1 TO MENU
```

```
330   LOCATE 25,I+3 : PRINT MENU$(I)
340 NEXT I
350 '
360 LOCATE 18,12
370 PRINT STRING$(40,149)
380 LOCATE 25,13
390 PRINT "ﾒﾆｭｰ ﾊﾞﾝｺﾞｳ ｦ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ"
400 N$=INKEY$ : IF N$="" THEN GOTO 400
410 N=VAL(N$)
420 IF N>MENU OR N=0 THEN BEEP : GOTO 400
430 '
440 COLOR 4
450 LOCATE 25,N+3
460 PRINT MENU$(N)
470 COLOR 0
480 FOR I=1 TO 100 : BEEP 1 : BEEP 0 : NEXT I
490 LOCATE 20,13 : PRINT SPC(40)
500 COLOR 6 : LOCATE 27,13
510 PRINT "  ｼﾞｯｺｳ [,]  ｷｬﾝｾﾙ [.]  "
520 COLOR 0
530 I$=INKEY$ : IF I$="" THEN GOTO 530
540 IF I$="," AND N=MENU THEN CLS : END
550 IF I$="," THEN RUN PROG$(N)
560 IF I$<>"." THEN BEEP : GOTO 530
570 LOCATE 25,N+3 : PRINT MENU$(N)
580 LOCATE 20,13 : PRINT SPC(40)
590 GOTO 380
600 '
610 '----------------------------------------
620 ' menu data
630 '----------------------------------------
640 DATA 6
650 DATA ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ ···············1,DMAIL1.BAS
660 DATA ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ ················2,DMAIL2.BAS
665 DATA ﾈﾝﾚｲ ﾉ ｺｳｼﾝ ·················3,DMAIL3.BAS
670 DATA ｱﾃﾅ ﾉ ｲﾝｻﾂ ··················4,DMAIL4.BAS
680 DATA ﾏｽﾀｰ ﾒﾆｭｰ ﾍ ﾓﾄﾞﾙ ··········5,MSMENU
690 DATA ｵﾜﾘ ························6,dummy
```

第3章
マスターメニューと
オートスタート

3・1　万能マスターメニュープログラム

これまでの宛て名印刷プログラムの勉強、いかがでしたか？

宛て名印刷ばかりでちょっとさえない、とお考えの方も多いかと思います。しかし、ディスク操作をマスターするには宛て名管理は最適の題材といえます。

このテーマ自体がどなたからもそれほどかけはなれた問題ではなく、また内容がそれほど複雑でもなく、しかしそれにもかかわらずディスク操作に必要な要素はひととおりもっているからです。

ここまでが理解できると、あとはずいぶん楽になると思います。

さて話が脱線しましたが、私たちはこれまで2つのメニュープログラム

menu1.bas
menu2.bas

↓図❶



を作ってディスクにセーブしてきました。

この節では、これらのメニュープログラムを管理する**マスターメニュープログラム**を作っていこうというわけです。いや、作るというより、ご紹介するといったほうが当たっているかもしれません。というのは本節では、プログラムの作り方は説明せず、プログラム総リストをかかげるにとどめたからです。

これまでのプログラムと原理的には変わりありませんので、ステートメントを追っていただけば、たぶんご理解いただけると思います。

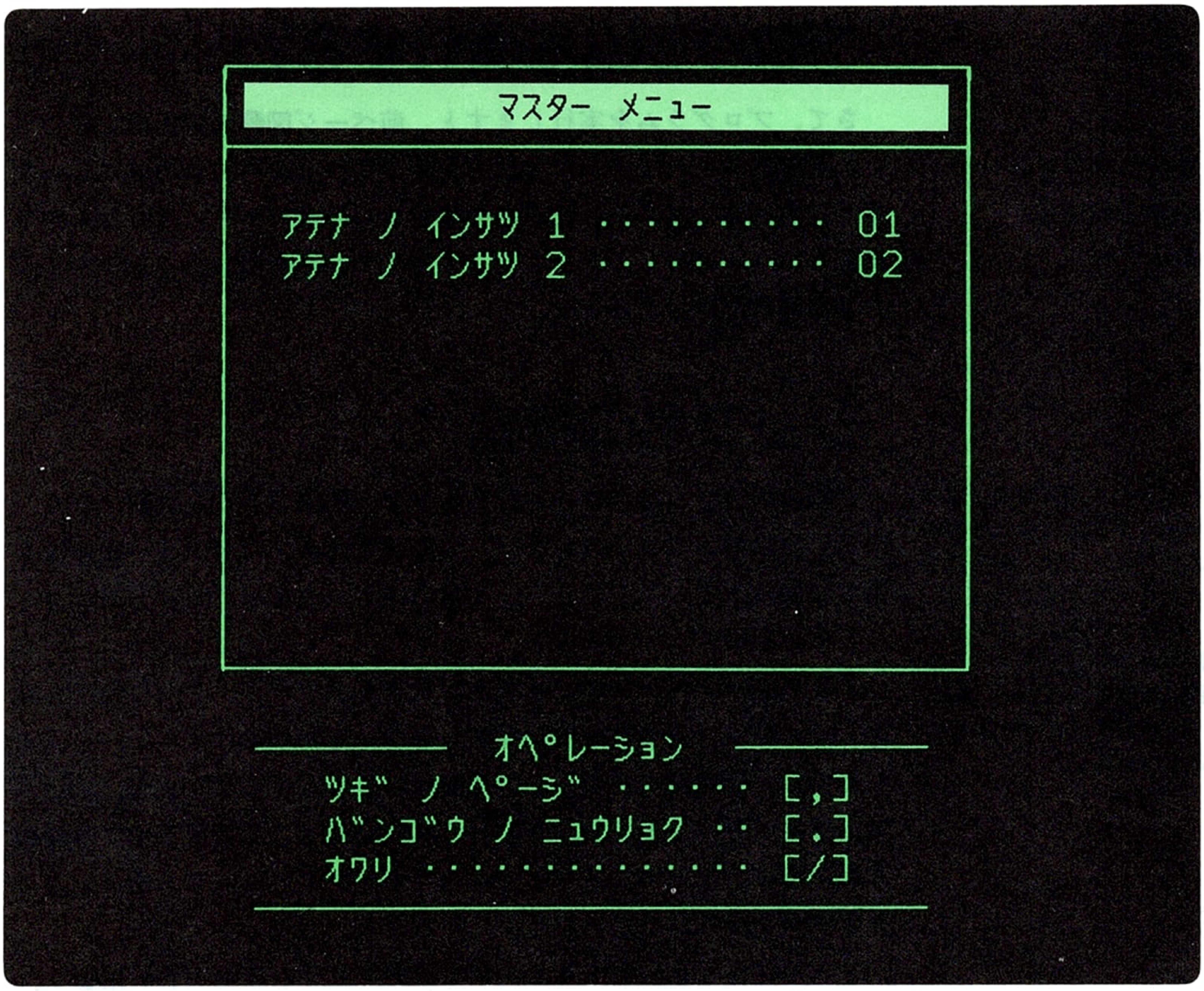
私たちの作った2つのメニュープログラムを仮りに

アテナ　ノ　インサツ　1
アテナ　ノ　インサツ　2

としておきます（実際には**「宛て名管理1」「宛て名管理2」**のほうがよかったかと思います）。これら2つのメニューを画面に表示させ、キー操作だけで実行できるようにしようというわけです。

図❶の画面をごらんください。これがマスターメニュープログラムの**画面レイアウト**です。

図❷⬇



〈注〉どんなに数がふえても使えるということで「万能」としましたが、ディスク容量に制限があるので、おのずと限定されます。本節でのプログラムはメニュー番号指定を2桁にしてありますので、最高99になります。これを3桁にすれば、もっと幅広く使用できます。

周囲が緑の罫線で囲まれていて、その中に**緑のアミの部分**がありますね。ここに**仕事名（ジョブ名）**を表示しますが、10個まで入れることができます。

私たちの仕事名はまだ2つしかありませんが、このマスターメニュープログラムは、**どんなに仕事名がふえても使える**(注)ようにしてあります。つまり、キー操作だけで次々にページを変えることができ、好きな仕事を実行できますので大変便利です。

図❶の下のほうをごらんください。ケースバイケースの**キーオペレーション**が表示されます。

なお、仕事を終らせたいときは [/] キー（スラッシュキー：テンキー側の割り算記号)、ページを変えてメニューを表示したいときは、テンキー側の [,] (コンマ)キー、メニュー番号入力の場合にはやはりテンキー側の [.] (小数点)キーを押します。

また、図❶の画面の左側にタテにふってある数字は、プログラムを実行したときの表示ではありません。みなさんがプログラムを入力するときに便利なように、Y座標位置を入れておきました。

プログラムを実行すると……

次ページのメニュー
仕事名の指定
ジョブエンド

さて、プログラムを実行しますと、前ページ図❷の画面のようにメニューが表示されます。いま述べたように、**次のページのメニューを見たければコンマキー、仕事名の指定なら小数点キー、終りなら割り算記号のキー**を押してください。すべてテンキー側にありますから、右手の操作だけですみます。

ジョブ名の指定

〈注〉メニュー番号の入力はINPUT＄関数による2文字入力でキャッチしますので2桁で入力するようにしてください。なお、入力した数字はINPUT文のように画面に表示されません。また、リターンキーを押す必要もありません。

図❷の画面で小数点キーを押しますと、図❸のような画面になり、指定した仕事名のところが反転して表示されます（**メニュー番号は2桁の数字で入力**してください。2番のメニューなら「02」というようにです)。(注)

そしてキーオペレーション表示が変わりますから、これに従って操作してください。**ゼロキー**を押すともう一度初めからメニューを表示しますし、**小数点キー**ならその画面での番号入力になります。また、**コンマキー**を押しますと、198ページの図❹のように、**ドライブ番号の入力**を求めてきます。プログラムをロードするためのファイルディスクリプタを作るためです。

ドライブ番号を入力すると、すぐにプログラムのロードが始まり、

その後実行されて、目的のプログラムのメニューが表示されます。

ここで何か処理を行い、再びマスターメニューへ戻ることもできます。マスターメニューでスラッシュキーを押すとジョブエンドとなり、画面は図❺のようになります。なぜか、ここだけ片こと英語でメッセージが表示されています。

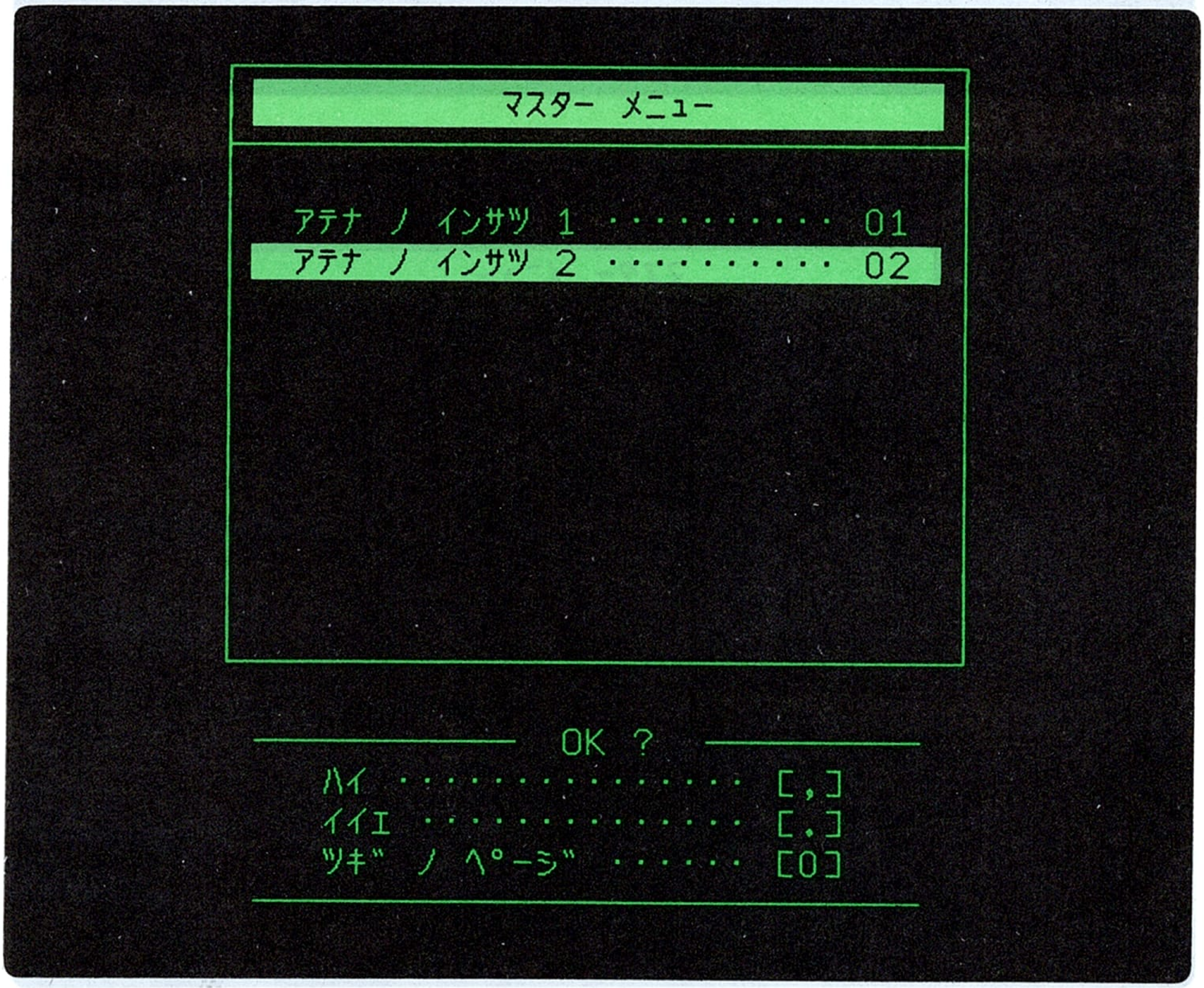
以上が**操作方法の説明**です。かなり便利にできていますので、ぜひご活用ください。

プログラム総リストを200ページにのせておきました。

なお、**メニューを追加したいときは**、1770行以降のDATA文を入力すれば、いくつでも登録できます。ただしこの場合、必ず最後に、「END」というデータを入れておいてください。

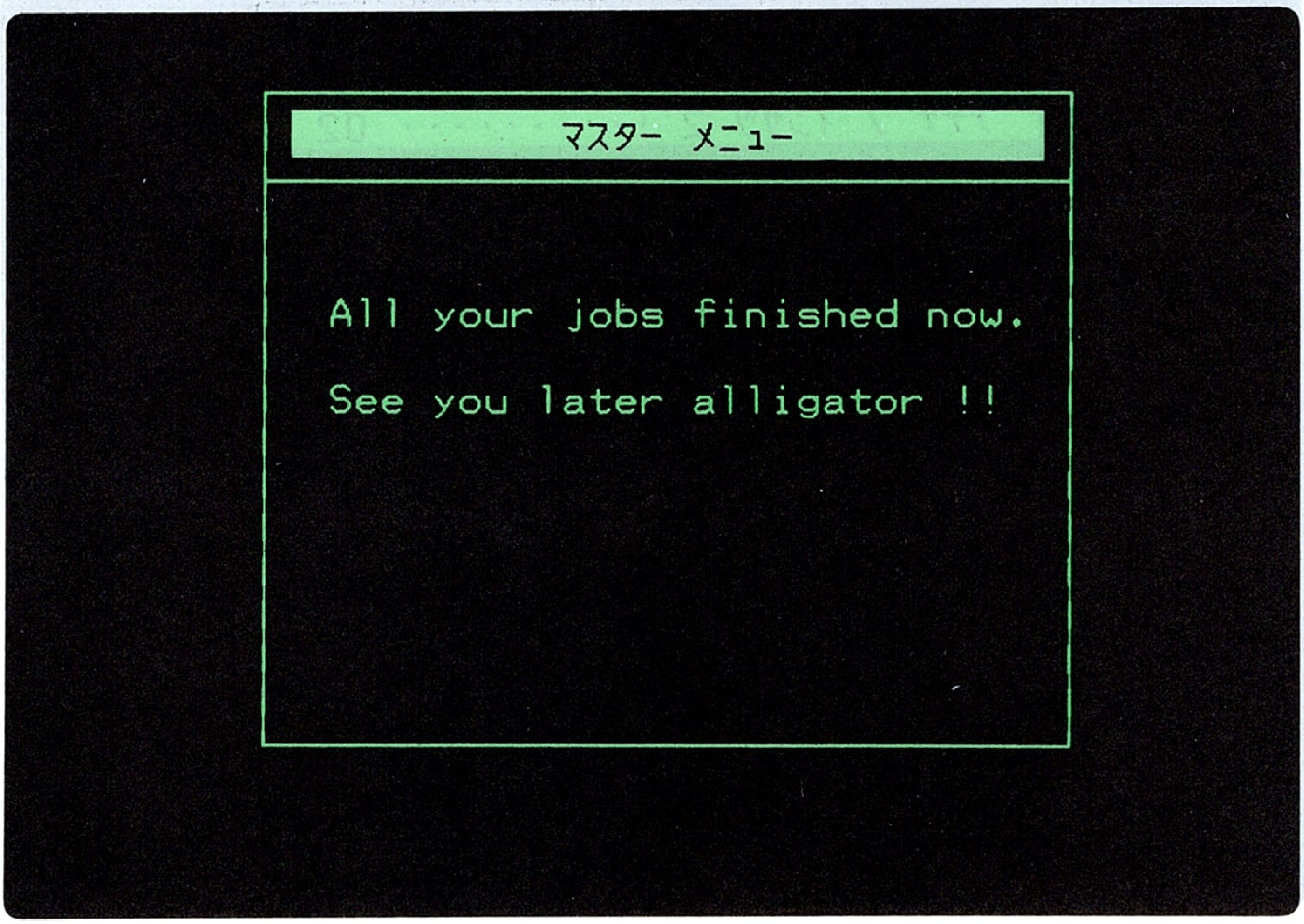
また、本書での**プログラムの流れ（フロー）**を199ページに示しましたので、これも参考にしてください。

図❸⬇

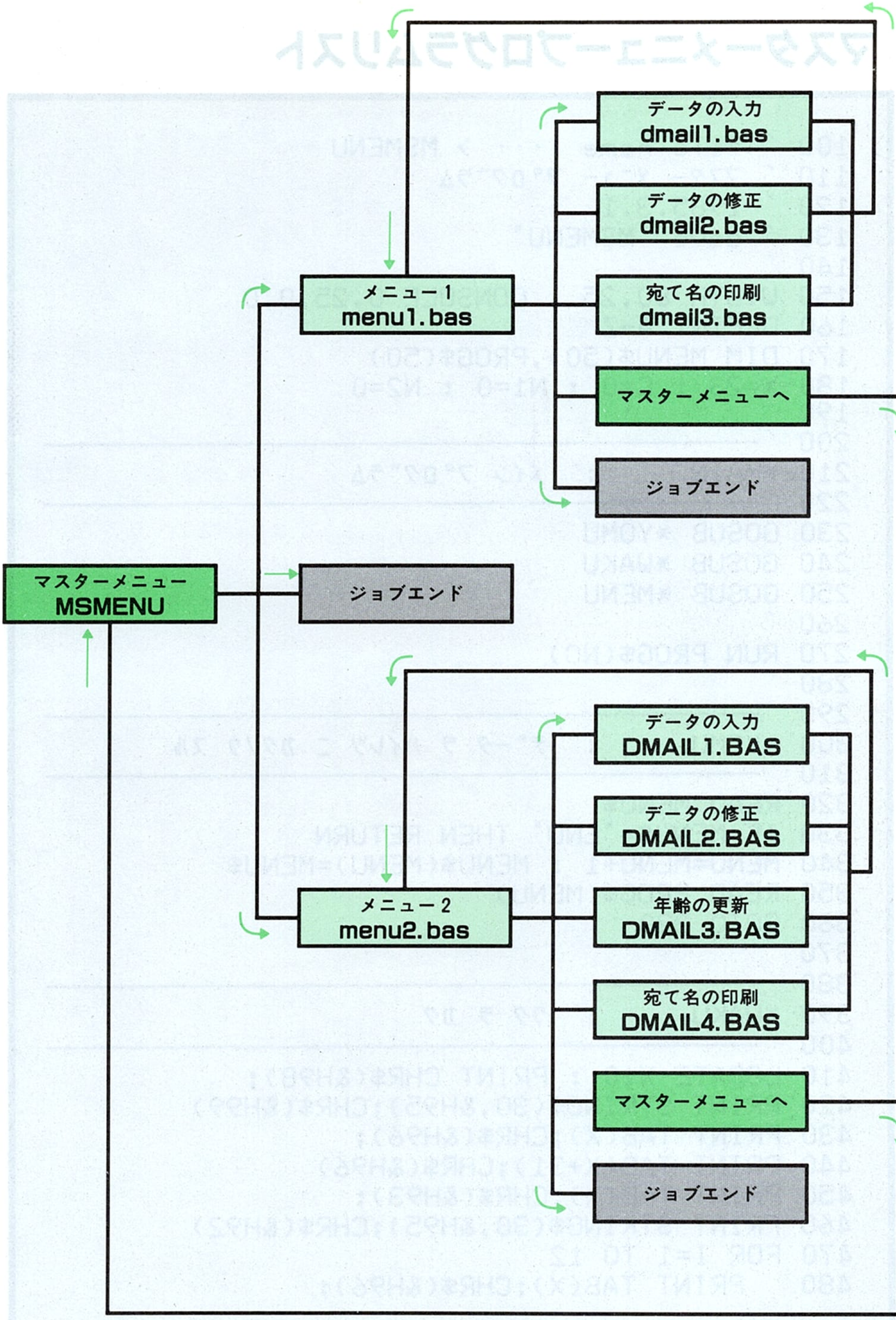



マスター メニュー
アテナ ノ インサツ 1 ·········· 01
アテナ ノ インサツ 2 ·········· 02
ドライブ バンゴウ ヲ ドウゾ

↑図❹


マスター メニュー
All your jobs finished now.
See you later alligator !!

↑図❺



マスターメニュー
MSMENU
ジョブエンド
メニュー1
menu1.bas
データの入力
dmail1.bas
データの修正
dmail2.bas
宛て名の印刷
dmail3.bas
マスターメニューへ
ジョブエンド
メニュー2
menu2.bas
データの入力
DMAIL1.BAS
データの修正
DMAIL2.BAS
年齢の更新
DMAIL3.BAS
宛て名の印刷
DMAIL4.BAS
マスターメニューへ
ジョブエンド

マスターメニュープログラムリスト

```
100 '  file name ････ > MSMENU
110 '  ﾏｽﾀｰ ﾒﾆｭｰ ﾌﾟﾛｸﾞﾗﾑ
120 '  1985.3.1
130 '  save "MSMENU"
140 '
150 WIDTH 80,25 : CONSOLE 0,25,0,0
160 DEFINT A-Z
170 DIM MENU$(50),PROG$(50)
180 X=23 : C=0 : N1=0 : N2=0
190 '
200 '------------------------------------------------------------
210 *MAIN        :' ﾒｲﾝ ﾌﾟﾛｸﾞﾗﾑ
220 '------------------------------------------------------------
230 GOSUB *YOMU
240 GOSUB *WAKU
250 GOSUB *MENU
260 '
270 RUN PROG$(NO)
280 '
290 '------------------------------------------------------------
300 *YOMU        :' ﾃﾞｰﾀ ｦ ﾊｲﾚﾂ ﾆ ｶｸﾉｳ ｽﾙ
310 '------------------------------------------------------------
320 READ MENU$
330 IF MENU$="END" THEN RETURN
340 MENU=MENU+1 : MENU$(MENU)=MENU$
350 READ PROG$(MENU)
360 GOTO 320
370 '
380 '------------------------------------------------------------
390 *WAKU        :' ﾜｸ ｦ ｶｸ
400 '------------------------------------------------------------
410 LOCATE X,0 : PRINT CHR$(&H98);
420 PRINT STRING$(30,&H95);CHR$(&H99)
430 PRINT TAB(X);CHR$(&H96);
440 PRINT TAB(X+31);CHR$(&H96)
450 PRINT TAB(X);CHR$(&H93);
460 PRINT STRING$(30,&H95);CHR$(&H92)
470 FOR I=1 TO 12
480   PRINT TAB(X);CHR$(&H96);
```

```
490   PRINT TAB(X+31);CHR$(&H96)
500 NEXT I
510 PRINT TAB(X);CHR$(&H9A);
520 PRINT STRING$(30,&H95);CHR$(&H9B)
530 LOCATE X+3,1 : COLOR 4
540 PRINT SPC(5);"     マスター メニュー     ";
550 PRINT SPC(5) : COLOR 0
560 RETURN
570 '
580 '────────────────────────────────────────
590 *MENU       :' メニュー ヲ ヒョウシ゛ スル
600 '────────────────────────────────────────
610 FOR I=1 TO MENU STEP 10
620   FOR J=1 TO 10
630     C=C+1 : N2=C
640     IF C>MENU THEN OWARI=-1 : N1=C-J+1
          : N2=C-1 : J=11 : GOTO 660
650     LOCATE X+3,J+3 : PRINT MENU$(C)
660   NEXT J
670   IF OWARI THEN GOTO 690
680   N1=C-J+2
690   LOCATE X+2,17
700   PRINT "────────  オヘ゜レーション  ────────"
710   LOCATE X+5,18
720   PRINT "ツキ゛ ノ ヘ゜ーシ゛ ・・・・・・ [,]"
730   LOCATE X+5,19
740   PRINT "ハ゛ンコ゛ウ ノ ニュウリョク ・・ [.]"
750   LOCATE X+5,20
760   PRINT "オワリ ・・・・・・・・・・・・・・ [/]"
770   LOCATE X+2,21
780   PRINT "──────────────────────────────"
790   GOSUB *BUZZER
800   I$=INKEY$ : IF I$="" THEN GOTO 800
810   IF I$="," THEN GOTO 850
820   IF I$="." THEN GOTO 930
830   IF I$="/" THEN GOTO 1630
840   BEEP : BEEP : GOTO 800
850   GOSUB *KESU1
860 NEXT I
870 '
880 C=0 : N1=0 : N2=0 : OWARI=0 : GOTO *MENU
890 '
```

```
900 '────────────────────────────────────
910 ' ﾒﾆｭｰ ﾊﾞﾝｺﾞｳ ﾉ ﾆｭｳﾘｮｸ
920 '────────────────────────────────────
930 GOSUB *KESU2
940 LOCATE X+8,17
950 PRINT "ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ";
960 NO$=INPUT$(2) : NO=VAL(NO$)
970 LO=(NO-1) MOD 10+1
980 IF NO<N1 THEN BEEP : GOTO 940
990 IF NO>N2 THEN BEEP : GOTO 940
1000 LOCATE X+2,LO+3 : COLOR 4
1010 PRINT " ";MENU$(NO);" " : COLOR 0
1020 GOSUB *BUZZER
1030 LOCATE X+2,17
1040 PRINT "──────────  OK ?  ────────"
1050 LOCATE X+5,18
1060 PRINT "ﾊｲ ............... [,]"
1070 LOCATE X+5,19
1080 PRINT "ｲｲｴ ............... [.]"
1090 LOCATE X+5,20
1100 PRINT "ﾂｷﾞ ﾉ ﾍﾟｰｼﾞ ...... [0]"
1110 LOCATE X+2,21
1120 PRINT "──────────────────────────"
1130 I$=INKEY$ : IF I$="" THEN GOTO 1130
1140 ON INSTR("0,.",I$) GOTO 1200,1260,1360
1150 BEEP : BEEP : GOTO 1130
1160 '
1170 '───────────────────────────────────
1180 ' ﾓｳ ｲﾁﾄﾞ ﾒﾆｭｰ ｦ ﾋｮｳｼﾞｽﾙ
1190 '───────────────────────────────────
1200 C=0 : N1=0 : N2=0 : OWARI=0 : GOSUB *KESU1
1210 GOSUB *KESU2 : GOTO *MENU
1220 '
1230 '───────────────────────────────────
1240 ' ﾌｧｲﾙ ﾃﾞｨｽｸﾘﾌﾟﾀ ｦ ﾂｸﾙ
1250 '───────────────────────────────────
1260 GOSUB *KESU2 : GOSUB *BUZZER
1270 LOCATE X+4,17
1280 PRINT "ﾄﾞﾗｲﾌﾞ ﾊﾞﾝｺﾞｳ ｦ ﾄﾞｳｿﾞ";
1290 DRIVE$=INPUT$(1)
1300 PROG$(NO)=DRIVE$+":"+PROG$(NO)
1310 RETURN
```

```
1320 '
1330 '────────────────────────────────────────
1340 '  ﾌﾀﾀﾋﾞ ﾊﾞﾝｺﾞｳ ﾆｭｳﾘｮｸ
1350 '────────────────────────────────────────
1360 LOCATE X+2,LO+3
1370 PRINT "  ";MENU$(NO);"  "
1380 GOTO 930
1390 '
1400 '────────────────────────────────────────
1410 *BUZZER
1420 '────────────────────────────────────────
1430 FOR K=1 TO 100 : BEEP 1 : BEEP 0 : NEXT K
1440 RETURN
1450 '
1460 '────────────────────────────────────────
1470 *KESU1
1480 '────────────────────────────────────────
1490 FOR K=4 TO 13
1500   LOCATE X+2,K : PRINT SPC(28)
1510 NEXT K
1520 RETURN
1530 '
1540 '────────────────────────────────────────
1550 *KESU2
1560 '────────────────────────────────────────
1570 FOR K=17 TO 21
1580   LOCATE X,K : PRINT SPC(30)
1590 NEXT K
1600 RETURN
1610 '
1620 '────────────────────────────────────────
1630 '  ｼﾞｮﾌﾞ ｴﾝﾄﾞ
1640 '────────────────────────────────────────
1650 GOSUB *KESU1 : GOSUB *KESU2
1660 GOSUB *BUZZER
1670 LOCATE X+3,5
1680 PRINT "All your jobs finished now."
1690 LOCATE X+3,7
1700 PRINT "See you later alligator !!"
1710 LOCATE 0,22
1720 END
1730 '
```

```
1740 '―――――――――――――――――――――――
1750 ' ﾃﾞｰﾀ
1760 '―――――――――――――――――――――――
1770 DATA ｱﾃﾅ ﾉ ｲﾝｻﾂ 1 ・・・・・・・・・・ 01
1780 DATA menu1.bas
1790 DATA ｱﾃﾅ ﾉ ｲﾝｻﾂ 2 ・・・・・・・・・・ 02
1800 DATA menu2.bas
1810 DATA END
```

3･2　オートスタート

オートスタートとは

私たちはいままで、システムを立ち上げるとき、

How many files (0−15) ?

という問いかけに対して「3」とか「5」とかを入力してリターンキーを押してきました。

そして「Ok」の表示がでたら、必要なプログラムをロードして実行してきました。

オートスタートとは、これらの操作をまったくしなくても、自動的にプログラムを実行してくれるスタート方法をいいます。

ここでは、**FRモデル30**を例にとって説明します。

ユーティリティプログラムのロード

それでは、オートスタートをセットする作業にとりかかりますが、このプログラムは、

dskut2. J 88

の中に組み込まれています。これをロードしてからRUNさせてください。

するとメニュー画面が表示されますので、4番の「**IDセクタの書き換え**」を選びます。

これがオートスタートさせるためのプログラムです。

4をキーインしてからリターンキーを押してください。

するとプログラムが実行され、**「システムをセットしたドライブ番号」**を入力するよう問いかけてきます。**システムのセーブしてないディスクでは、この処理をしてもオートスタートにはなりません**ので注意してください。

さて、私たちがこれまで一連のプログラムをセーブしてきたディスクを**ドライブ1**にセットして「**1** ⏎」としてください。

すると次に、

同時にオープンする最大ファイル数（0−15，設定しないときは−1）？

と聞いてきます。

これに対しては立ち上げ時の「How many files (0 −15) ?」のメッセージに対して入力した値をキーインします。

たとえば「**5** ↵ 」としてみましょう。

今度は「**カレンダ時計の西暦年**」をたずねてきます。すでにセットしてある西暦でよければ「**−1**」を、変更したければ西暦の下2桁をキーインして ↵ キーを押します。

たとえば1986年なら「**86** ↵ 」とします。

すると次に、

実行したいBASIC命令を入力してください。

と聞いてきます。

私たちはマスターメニュープログラムを実行したいので、

run "MSMENU" ↵

と入力します。

これでオートスタートのためのデータ入力は終りです。

以上の操作が終ると、先に入力した内容でよいかどうか確認してきます。もし間違いなければ「y ↵ 」、再入力したければ「n ↵ 」とします。

yを入力するとIDセクタの書き換えが行なわれ、完了すると「終了しました」のメッセージが表示されると同時に「ディスクユーティリティの作業を続けるかどうか」をたずねてきます。ここではnを入力して作業を終りましょう。

さあ、これでオートスタートのセットができました。リセットボタンを押してためしてみてください。自動的にメニュープログラムが実行され、メニュー画面が表示されたでしょう。

第4章
日本語版 簡易住所録

4·1 プログラミングの前に

私たちはこれまで、データファイルへの書き込みデータをすべて**カタカナ**あるいは**英数字**として扱ってきました。

本章では、**N88日本語BASICシステムディスク**を使って漢字データの扱いを勉強してみましょう。取り上げるテーマにはいろいろ考えられますが、一番需要の多い**「住所録」**を選ぶことにしました。いままでアドレスブックに書いていた住所録を、今度はコンピュータにやらせてみようというわけです。

住所録プログラムもこるときりがありませんが、ここでは原理を覚えていただくことが目的ですので、ごく簡単なプログラムを紹介することにします。

データディスクを作る

本書ではプログラムファイル用ディスクはドライブ1に、データファイル用ディスクはドライブ2にセットしてディスク操作を行ってきましたが、これから私たちが使うN88日本語BASICシステムは辞書ファイルその他でかなりの容量を使用しています。そこで本章では考え方を変えて、ドライブ1にはN88日本語BASICシステムディスクを、そしてドライブ2側のディスクはプログラムファイル、データファイル兼用として使用することにします。

つまり、新しいディスクを1枚用意してこれからデータディスクを作り、これをドライブ2にセットして使用することにします。

データディスクの作成は

dskut2．J88

によって行いますので、まず**「N88BASICシステムディスク」**をドライブ1にセットし、上記のユーティリティプログラムを実行してください。

メニュー画面が現れたら**「データディスクの作成」**を選択し、メッセージに従って新しい未使用のディスクからデータディスクを作ってください。

次に、いま作成したデータディスクに「dskut 2．j88」をセーブしておきます。ユーティリティプログラムはディスクの管理に使う機会が多いのでセーブしておくと便利です。

ユーティリティプログラムdskut 2．j88を動作させるためには、これ

と対をなすマシン語プログラム、

dskut2＊bin

が必要です。

しかし、マシン語プログラムをセーブするには普通のSAVEコマンドは使えませんので、ファイル転送プログラム

×files. J88

を実行して行います。

この方法については、付属の「**ユーティリティマニュアル**」の「**ファイルの転送**」の項にくわしく説明されていますのでそちらを参照してください。

いま作った「データディスク」に「**dskut 2 ＊bin**」を転送できたら準備完了です。

N88日本語BASICの立ち上げ

それでは次に「N88日本語BASIC」のモードにしておきましょう。「**N88日本語BASICシステムディスク**」をドライブ1に、いま作成したデータディスクをドライブ2にセットしてからリセットボタンを押してください。

しばらくして「How many files（0－15）?」のメッセージが表示されますから、これに対しては「**5** ⏎」としてください。すると画面に「N88日本語BASIC」の表示がでて立ち上げが終了します。

なお、日本語を使うためには、常に「N88日本語BASICシステムディスク」がドライブ1にセットしてあることが条件となりますので注意してください。「N88BASICシステムディスク」では動作しません。

4・2 2バイト系文字列について

前章までのデータファイルにおいて私たちは、データとして数値あるいは1バイト系の文字列を扱ってきました。

1バイト系の文字とは、キャラクタコード表に示される文字のことで、たとえばアルファベットの「A」は16進数の「41」という1バイトのコードで表現できます。

これに対して**2バイト系の文字**とは、リファレンスマニュアルの巻末に掲載されている「**日本語コード表**」に示される文字のことで、たとえばアルファベットの「A」は16進数の「8260」という2バイトのコードで表わされます。

「日本語コード表」を見るとわかるように、たとえば2バイト系のAに対しては

8260
2341

という2つのコードが対応しています。

上のコードは「**シフトJISコード**」と呼ばれるもので、PC-8801MKII FR/MRではこのコード体系が採用されています。

もう一方の「2341」というコードは「**JISコード**」と呼ばれ、これはPC-9801系の漢字処理に使われています。

詳しい説明は省略しますが、JISコードの場合、2バイト系の漢字データであることをチェックするために前後に「**漢字イン**」「**漢字アウト**」の識別データを付加しなければなりませんが、シフトJISコードにおいてはその必要がないため、ディスクおよびメモリを有効に使うことができます。

とにかくFR/MRのN88日本語BASICで漢字を扱う場合、そのコード体系はシフトJISコードによる、と覚えておいてください。

なお、N88日本語BASICではなくN88BASICシステムディスクで立ち上げた場合の漢字コードはJISコードによっていますので注意してください。

ではここで、簡単なプログラムを作り、ディスク上で1バイト系文字列、2バイト系文字列がどんなふうに記録されているか調べてみましょう。

テストのためのサンプルプログラムを右ページに示しますのでキーインしてください。

```
100 OPEN "2:test" AS #1
110 FIELD #1,30 AS A$
120 INPUT "好きな言葉をどうぞ  ",B$
130 NAGASA=LEN(B$)
140 LSET A$=B$
150 PUT #1,1
160 CLOSE
170 OPEN "2:test" FOR INPUT AS #1
180 FOR I=1 TO NAGASA
190   C$=INPUT$(1,#1)
200   C$=HEX$(ASC(C$))
210   PRINT C$" ";
220 NEXT I
230 END
```

プログラムの意味は、次の通りです。

- **100行**――ランダムファイルをオープンします。
- **110行**――フィールドの割り当てをします。A＄という文字列に30バイトを割り当てています。
- **120行**――文字列の入力をします。
- **130行**――入力した文字列の長さを変数NAGASAに代入します。
- **140行**――キー入力された文字列をバッファ上でセットします。
- **150行**――ランダムファイルに書き込みます。
- **160行**――ファイルを閉じます。
- **170行**――ランダムファイルを今度はシーケンシャルファイルとして扱い、読み出しモードでオープンします。
- **180行**――先ほど調べた文字列の長さだけFOR～NEXTを組みます。
- **190行**――INPUT＄関数により1番バッファから1文字読み出します。
- **200行**――いま読み出した1文字をアスキーコードに変換し、それをさらにHEX＄関数で16進数に変換します。
- **210行**――16進数を画面に出力します。

それでは、このプログラムを実行してテストしてみましょう。
実行結果の一例を次ページに示します。



この例では、「好きな言葉をどうぞ」の入力促進に対して、まず、
f･1 キーを押して日本語入力モードにし

百科辞典

を入力しました。つまり、2バイト系文字を4文字入力したことになります。

次に、再び f･1 キーを押して日本語入力モードから抜け出し、今度はカタカナで「デス」を入力します。この場合濁点も1文字として数えられますから、計3文字の1バイト系文字列をキーインしたことになります。

入力が終り ⏎ キーを押すと、いま入力された文字列が16進数に変換されて画面に表示されます。

日本語コード表をごらんください。

最初の2バイト「**95　53**」は漢字の「百」に、「**89　C8**」は「科」に、「**8E　AB**」は「辞」に、「**93　54**」は「典」に対応しているのがわかります。

次に今度は**キャラクタコード表**をごらんください。

「**C3**」は「テ」に、「**DE**」は濁点に、「**BD**」は「ス」に対応しているのを知ることができます。

このことからもわかるように、「百科辞典デス」はランダムファイル上には、

95 53 89 C8 8E AB 93 54 C3 DE BD

というイメージで記録されているのがわかります。全部で11バイトです。「百科辞典」は2バイト×4文字で8バイト、「デス」が3バイトで計11バイトということになります。

つまり、ディスク上では日本語は2バイトが必要であり、1バイト系のカタカナ・英数字は1バイトあれば十分ということになります。この原理が、フィールド設定のときに大切になりますので覚えておいてください。

4・3　新規登録プログラム

この住所録プログラムで使用するデータファイルは次の3つとします。

❶　サクイン1

コード用のインデックスファイルです。これは従来どおりシーケンシャルファイルとし、数値で書き込みます。

❷　サクイン2

これもやはりシーケンシャルファイルですが、これは氏名の先頭の1文字（漢字）を書き込んでおき、頭文字による検索のためのインデックスファイルとします。

❸　ジュウショ

住所登録用のランダムファイルです。

フィールドの割り当て

ランダムファイル「**ジュウショ**」には氏名、郵便番号、住所1、住所2、電話番号の5項目を書き込むこととし、各項に次のようにフィールドを割り当てます。

- 氏名――――20バイト
- 郵便番号―― 7バイト
- 住所1――――40バイト
- 住所2――――40バイト
- 電話番号――15バイト

全部で122バイト使用し、残り134バイトは未使用とします。あとでみなさんが他の項目を作って使用してください。

ところで各項に割り当てる文字の型ですが、郵便番号および電話番号には1バイト系の文字列を使い、そのほかの項目についてはすべて2バイト系の漢字を使うことにします。

もちろん2バイト系、1バイト系の混在も許されますが、このプログラムでは分けておくことにします。

つまり、上のフィールド割り当てでいくと、氏名には漢字を10文字まで、住所1および住所2には漢字を20文字まで入力できます。

また、郵便番号、電話番号はキャラクタでそれぞれ7、15文字まで入力できます。

初期設定とファイルオープン

```
100 ´ 住所録プログラム
110 ´ save "2:adres1.bas"
120 ´ 新規登録
130 ´
140 *SHOKISETTEI
150  SCREEN 4
160  WIDTH 80,20 : CONSOLE 0,20,1,0
170  DEFINT A-Z
180  DIM CODE(500)
190 ´
200 *JYUNBI
210  GOSUB *FILE.OPEN
220  IF LOF(1)<>0 THEN GOSUB *CODE.YOMU : GOTO 260
230  OPEN "2:ｻｸｲﾝ1" FOR OUTPUT AS #2 :´ コード索引
240  OPEN "2:ｻｸｲﾝ2" FOR OUTPUT AS #3 :´ 氏名索引
250  SEC=1 : GOSUB *MIDASHI : GOTO *NYUURYOKU
260  OPEN "2:ｻｸｲﾝ1" FOR APPEND AS #2
270  OPEN "2:ｻｸｲﾝ2" FOR APPEND AS #3
280  GOSUB *MIDASHI
```

プログラムは前に述べたようにドライブ2側のディスクにセーブしますので、110行のファイルディスクリプタには「2：」の指定が必要です。

N88日本語BASICシステムディスクで立ち上げるとSCREEN文の第1パラメータは自動的に「4」に設定されますが、他のプログラムを実行したあとでこのプログラムを実行することもありますので、一応宣言をしておきます（150行）。

180行の配列CODEはコード索引用です。すでに登録ずみのコードを読み込み、コード重複のチェック用に使います。

210行がランダムファイル「**ジュウショ**」のオープンのためのサブルーチンへのジャンプです。

このサブルーチンを次に示しますので入力してください。

```
720 '
730 *FILE.OPEN
740  OPEN "2:ｼﾞｭｳｼｮ" AS #1
750  FIELD #1,20 AS NM$,7 AS PN$,40 AS AD1$,
          40 AS AD2$,15 AS TL$
760  RETURN
```

バッファ上での変数は次のように各項目に割り当てられています。

◉氏名———NM$
◉郵便番号——PN$
◉住所1———AD1$
◉住所2———AD2$
◉電話番号——TL$

このランダムファイルをオープンし、もしファイル長が0でなければ、すでに登録ずみのコードがあるわけですからシーケンシャルファイルを読みにいきます。次のプログラムを入力してください。

```
770 '
780 *CODE.YOMU
790  OPEN "2:ｻｸｲﾝ1" FOR INPUT AS #2
800  WHILE NOT EOF(2)
810    C=C+1              :' カウンタを1進める
820    INPUT #2,CODE(C)   :' レコードを1件読み出す
830  WEND
840  SEC=C+1              :' 書き込みセクタを決める
850  CLOSE #2
860  RETURN
```

これは、みなさんはおなじみのルーチンですね。説明の必要はないと思います。このサブルーチンから帰った場合には、シーケンシャルファイルは追加の**APPEND**モードでオープンしなければなりません。260行へジャンプさせます。

この場合以外は初回の登録になりますから、**OUTPUT**モードでオープンします。

みなさんは日本語入力になれているからミスはないと思いますが、f･1 キーによる日本語入力モード/ノーマルモードの切り換えをしっかりしてください。プログラムのコマンド、ステートメントを日本語入力モードでキーインしないようにしてください。

シーケンシャルファイルをオープンしたら今度は、ジョブ名表示のサブルーチンへジャンプします。

この部分のプログラムは次のとおりです。

```
870 '
880 *MIDASHI
890  LOCATE 10,1 : PRINT "【住所録への新規登録】"
900  LINE (72 ,50)-(536 ,50)
910  LINE (72,230)-(536,230)
920  RETURN
```

〈注〉890行の「【」の入力は SHIFT キーを押しながら f・4 キーを押すとコード入力になりますから「8 1 7 9 ⏎」とキーインします。

900行、910行は各項目入力行の上下に線を引くステートメントですが、N88日本語BASICモードではグラフィックキャラクタが使えませんので、これをLINE文によって行っています。

項目の入力

次は、各項目の入力となります。次のプログラムをキーインしてください。

```
290 '
300 *NYUURYOKU
310  LOCATE 40,4
320  PRINT "終りの時は9999"
330  LOCATE 12,4 : INPUT "コード    ",CD
340  IF CD=9999 THEN GOTO *OWARI
350  LOCATE 40,4 : PRINT SPC(20)
360  GOSUB *CODE.CHECK  :' コード重複かどうかのチェック
370  LOCATE 12,5 : INPUT "名前      ",NAMAE$
380  LOCATE 12,6 : INPUT "郵便番号  ",YUUBIN$
390  LOCATE 12,7 : INPUT "住所1    ",JUUSHO1$
400  LOCATE 12,8 : INPUT "住所2    ",JUUSHO2$
410  LOCATE 12,9 : INPUT "電話番号  ",DENWA$
420  GOSUB *KEY.OPE.HYOUJI
430  I$=INKEY$ : IF I$="" THEN GOTO 430
440  IF I$=CHR$(13) THEN GOTO *KAKU
450  IF I$="/" THEN GOTO *KESU
460  BEEP : BEEP : GOTO 430
```

入力の終りは、コードに対して「**9999**」が入力されたときとします。

350行は「終りの時は9999」のメッセージを消去するものです。

もしコードに対して「9999」が入力されれば終了処理へ、そうでなければ、すでに登録ずみのコードがあるかどうか、つまり**コード重複のチェック**を行います。

この部分のプログラムを次に示します。

```
930 '
940 *CODE.CHECK
950  S=SEARCH(CODE,CD) :' 重複コードがある`FBN
960  IF S=-1 THEN RETURN  :' 重複コードなし
970  FOR BUZZER=1 TO 200 : BEEP 1 : BEEP 0 : NEXT BUZZER
980  LOCATE 40,4 : PRINT "コード重複です！"
990  FOR TIMER=1 TO 2000 : NEXT TIMER :' タイマー
1000  LOCATE 22,4 : PRINT SPC(20) :' エラーメッセージ消去
1010  RETURN *NYUURYOKU
```

SEARCH関数を使って整数配列CODEの中に、いま入力されたコードCDがあるかどうかチェックします。この関数が－1をその値として返せば、重複コードはないことになりますのでサブルーチンから戻ります。

－1以外の数値が返されれば「**コード重複**」のエラーメッセージを表示して、もう一度コード入力のやりなおしです。

コード重複でなければ、「名前」以降の各項目を次々と入力していきます。

電話番号まで入力が終ったら「登録」か「キャンセル」かの確認を行います。次のサブルーチンをキーインしましょう。

```
1020 '
1030 *KEY.OPE.HYOUJI
1040  LOCATE 10,13
1050  PRINT "キー操作   登録【ＲＥＴ】  キャンセル【／】"
1060  RETURN
```

[⏎] キーが押されれば登録、[／] キーが押されればキャンセルです。テンキー側でのキー操作だけで事がたります。

ファイルへの書き込み

[⏎] キーのキャラクタコードは「13」です。440行をごらんください。I＄にCHR＄（13）が代入されれば [⏎] キーが押されたということで、次ページに示す書き込み部分へジャンプします。

490行では、漢字で入力された姓の最初の1文字を取り出し、それを変数HEAD＄に代入しています。

漢字1文字は2バイトで構成されますから、LEFT＄により取り出す数は「2」を指定すればよいことになります。

```
470 '
480 *KAKU
490  HEAD$=LEFT$(NAMAE$,2)    :' 姓 の 頭 文 字 を 取 り 出 す
500  WRITE #2,CD              :' コ ー ド の 書 き 込 み
510  WRITE #3,HEAD$           :' 頭 文 字 の 書 き 込 み
520  LSET NM$ =NAMAE$         :' 氏 名
530  LSET PN$ =YUUBIN$        :' 郵 便 番 号
540  LSET AD1$=JUUSHO1$       :' 住 所 1
550  LSET AD2$=JUUSHO2$       :' 住 所 2
560  LSET TL$ =DENWA$         :' 電 話 番 号
570  PUT #1,SEC               :' ラ ン ダ ム フ ァ イ ル に 書 く
580  CODE(SEC)=CD             :' 配 列 に 新 し い コ ー ド を 登 録
590  SEC=SEC+1                :' セ ク タ を 1 つ 進 め る
```

500行はシーケンシャルファイルへのコード番号の書き込み、510行は姓の先頭1字の書き込みですが、バッファ番号の指定を間違わないようにしてください。正しくデータが書き込まれません。

580行は新規登録データのコードを配列に登録します。

590行でセクタを1つ進め、次に示す入力行およびキーオペレーション表示の消去処理を行います。

```
600 '
610 *KESU
620  FOR I=4 TO 9
630    LOCATE 12,I : PRINT SPC(60) :' 入 力 行 の 消 去
640  NEXT I
650  LOCATE 10,13
660  PRINT SPC(50)     :' キ ー オ ペ レ ー シ ョ ン 消 去
670  GOTO *NYUURYOKU
```

画面に表示された文字はスペースを出力して消せばよいですね。
消去処理の次は再び入力ルーチンへ飛ばします。
最後に終了処理ルーチンを入力してプログラムの完成です。

```
680 '
690 *OWARI
700  CLOSE
710  RUN "2:adres4.bas"  :' メ ニ ュ ー へ 戻 る
```

```
100 ' 住所録プログラム
110 ' save "2:adres1.bas"
120 ' 新規登録
130 '
140 *SHOKISETTEI
150  SCREEN 4
160  WIDTH 80,20 : CONSOLE 0,20,1,0
170  DEFINT A-Z
180  DIM CODE(500)
190 '
200 *JYUNBI
210  GOSUB *FILE.OPEN
220  IF LOF(1)<>0 THEN GOSUB *CODE.YOMU : GOTO 260
230  OPEN "2:ｻｸｲﾝ1" FOR OUTPUT AS #2 :' コード索引
240  OPEN "2:ｻｸｲﾝ2" FOR OUTPUT AS #3 :' 氏名索引
250  SEC=1 : GOSUB *MIDASHI : GOTO *NYUURYOKU
260  OPEN "2:ｻｸｲﾝ1" FOR APPEND AS #2
270  OPEN "2:ｻｸｲﾝ2" FOR APPEND AS #3
280  GOSUB *MIDASHI
290 '
300 *NYUURYOKU
310  LOCATE 40,4
320  PRINT "終りの時は９９９９"
330  LOCATE 12,4 : INPUT "コード     ",CD
340  IF CD=9999 THEN GOTO *OWARI
350  LOCATE 40,4 : PRINT SPC(20)
360  GOSUB *CODE.CHECK  :' コード重複かどうかのチェック
370  LOCATE 12,5 : INPUT "名前      ",NAMAE$
380  LOCATE 12,6 : INPUT "郵便番号   ",YUUBIN$
390  LOCATE 12,7 : INPUT "住所１    ",JUUSHO1$
400  LOCATE 12,8 : INPUT "住所２    ",JUUSHO2$
410  LOCATE 12,9 : INPUT "電話番号   ",DENWA$
420  GOSUB *KEY.OPE.HYOUJI
430  I$=INKEY$ : IF I$="" THEN GOTO 430
440  IF I$=CHR$(13) THEN GOTO *KAKU
450  IF I$="/" THEN GOTO *KESU
460  BEEP : BEEP : GOTO 430
470 '
480 *KAKU
490  HEAD$=LEFT$(NAMAE$,2)    :' 姓の頭文字を取り出す
500  WRITE #2,CD          :' コードの書き込み
510  WRITE #3,HEAD$        :' 頭文字の書き込み
520  LSET NM$ =NAMAE$       :' 氏名
530  LSET PN$ =YUUBIN$      :' 郵便番号
540  LSET AD1$=JUUSHO1$     :' 住所１
550  LSET AD2$=JUUSHO2$     :' 住所２
560  LSET TL$ =DENWA$       :' 電話番号
570  PUT #1,SEC          :' ランダムファイルに書く
580  CODE(SEC)=CD         :' 配列に新しいコードを登録
590  SEC=SEC+1           :' セクタを１つ進める
600 '
610 *KESU
620  FOR I=4 TO 9
```

```
630    LOCATE 12,I : PRINT SPC(60) :´ 入力行の消去
640   NEXT I
650   LOCATE 10,13
660   PRINT SPC(50)     :´ キーオペレーション消去
670   GOTO *NYUURYOKU
680 ´
690 *OWARI
700   CLOSE
710   RUN "2:adres4.bas"    :´ メニューへ戻る
720 ´
730 *FILE.OPEN
740   OPEN "2:ｼﾞｭｳｼｮ" AS #1
750   FIELD #1,20 AS NM$,7 AS PN$,40 AS AD1$,
            40 AS AD2$,15 AS TL$
760   RETURN
770 ´
780 *CODE.YOMU
790   OPEN "2:ｻｸｲﾝ1" FOR INPUT AS #2
800   WHILE NOT EOF(2)
810     C=C+1                :´ カウンタを1進める
820     INPUT #2,CODE(C)     :´ レコードを1件読み出す
830   WEND
840   SEC=C+1                :´ 書き込みセクタを決める
850   CLOSE #2
860   RETURN
870 ´
880 *MIDASHI
890   LOCATE 10,1 : PRINT "【住所録への新規登録】"
900   LINE (72 ,50)-(536 ,50)
910   LINE (72,230)-(536,230)
920   RETURN
930 ´
940 *CODE.CHECK
950   S=SEARCH(CODE,CD) :´ 重複コードがある`FBN
960   IF S=-1 THEN RETURN   :´ 重複コードなし
970   FOR BUZZER=1 TO 200 : BEEP 1 : BEEP 0 : NEXT BUZZER
980   LOCATE 40,4 : PRINT "コード重複です!"
990   FOR TIMER=1 TO 2000 : NEXT TIMER :´ タイマー
1000   LOCATE 22,4 : PRINT SPC(20) :´ エラーメッセージ消去
1010   RETURN *NYUURYOKU
1020 ´
1030 *KEY.OPE.HYOUJI
1040   LOCATE 10,13
1050   PRINT "キー操作  登録【RET】 キャンセル【/】"
1060   RETURN
```

4・4　新規登録してみよう

それではさっそくいま作成したプログラムを実行してみますが、その前に**プログラムをセーブ**しておいてください。

登録するデータですが、次の３名とします。

- コード――**100**
- 氏名――**福島　実**
- 郵便番号――**962-03**
- 住所１――**福島県岩瀬郡岩瀬村**
- 住所２――**100番地**
- 電話番号――**111-111-1111**

- コード――**150**
- 氏名――**日本太郎**
- 郵便番号――**702**
- 住所１――**岡山市洲崎**
- 住所２――**1-2-3**
- 電話番号――**222-222-2222**

- コード――**200**
- 氏名――**東京花子**
- 郵便番号――**113**
- 住所１――**文京区千駄木**
- 住所２――**1-5-6**
- 電話番号――**333-333-3333**

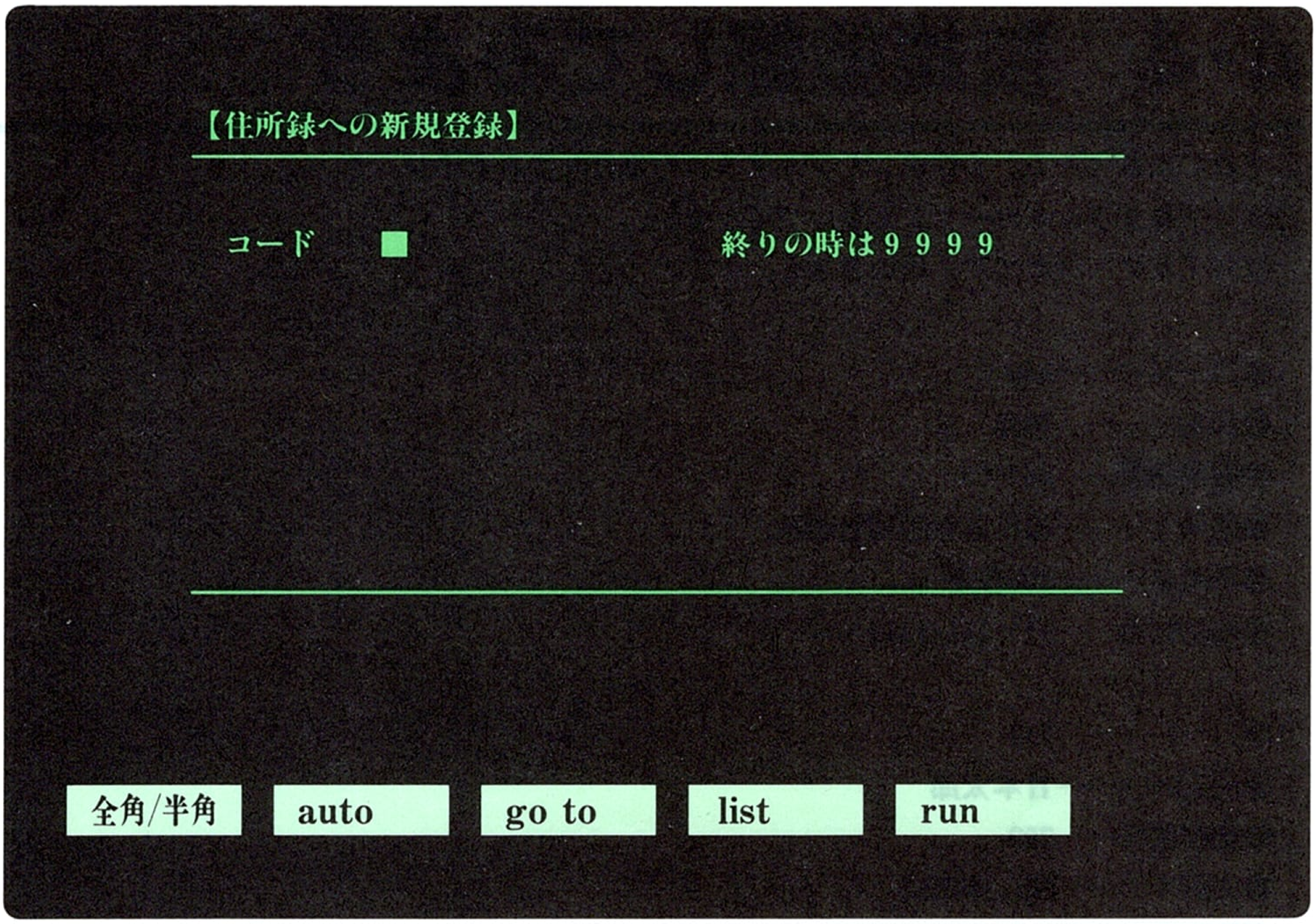
モードがN88日本語BASICになっていますね。確認できたらプログラムを実行してください。

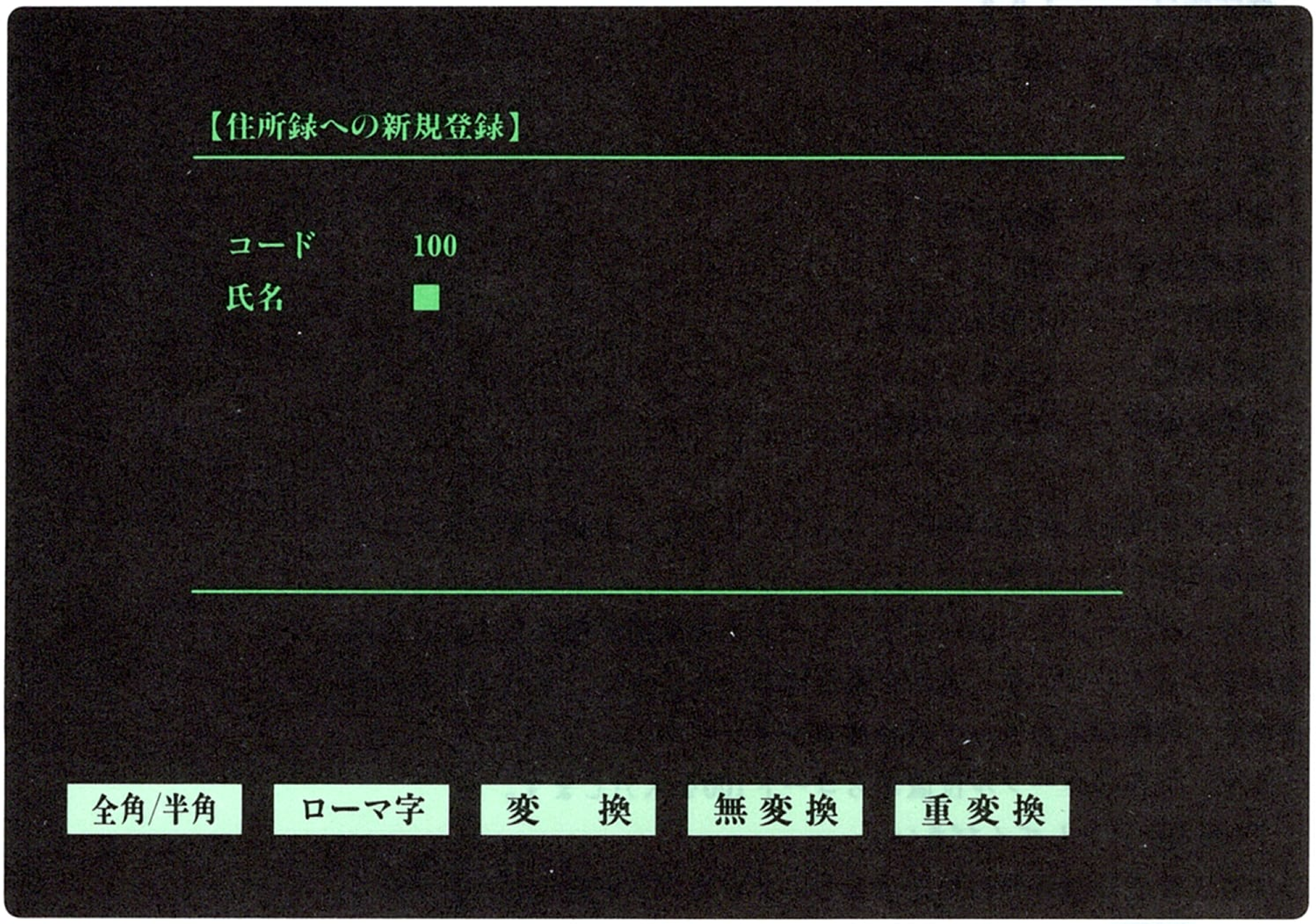
すると次ページ上の図のようにデータ入力画面が現れます。この画面はファンクションキー表示を見ればわかるように、１バイト系の**カタカナ・英数字入力モード（半角モード）**です。

まず、現在のカーソル位置からコード100を入力します。

「100 ⏎ 」としてください。

すると今度は改行されて氏名入力のプロンプトが表示されますから、ここで f･1 キーを押して**日本語入力モード（全角モード）**にします。このときの画面は次ページ下の図のようになっています。





現在「ローマ字入力モード」ですから カナ キーをロックし「**かな入力モード**」にします。これは、ファンクションキー表示の右端が

「〔**かな**〕」となるので確認できます。

かな入力モードになりましたら「福島」の読みである「フクシマ」をキーインします。

すると画面では「ふくしま」が反転表示されますから［f・3］キーを押して変換します。どうですか？　1回で「福島」と変換されましたね。

次に「実」の入力ですが、その前にスペースを入れておきますが、日本語入力モードではスペースキーは漢字変換キーとして使われますのでこのキーを押してもスペースを入力することはできません。

しかたないですから一度［f・1］キーを押して通常のモードに戻し、ここでスペースキーを2回たたいてスペースを入力します。このときのスペースは1バイト系ですから漢字1字分に合わせるためには2回たたけばよいですね。

スペースを入力したら［f・1］キーを押して再び日本語入力モードにします。そして「ミノル」とキーインして［f・3］キーかスペースキーをたたいて漢字変換です。この場合も1回で変換されて画面に、「実」と表示されましたね。では［↵］キーを押して決定してください。

これで氏名の入力は終りですのでもう一度［↵］キーを押します。

すると今度は郵便番号の入力です。郵便番号は1バイト系で入力しますので［f・1］キーを押して半角モードになおします。

そして「962－03［↵］」とキーインしてください。

次は住所の入力ですが［f・1］キーを押して全角モードにしてから同様に入力してください。

住所の入力が終ったら電話番号です。これは1バイト系ですから半角モードで入力します。

電話番号までの入力が終了すると**「登録するか」「キャンセルするか」**聞いてきます。

いま入力したデータで間違いなければ［↵］キーを、キャンセルなら［／］キーを押してもう一度やりなおしです。

［↵］キーを押して登録すると再びコード入力となりますから2つ目のデータをキーインしてください。

同様に3つ目のデータを入力し、4つ目のコード入力に対して9999［↵］とキーインして作業を終ります。するとファイルへのデータ書き込みが行なわれメニュープログラムのロードにかかります。しかしメニュープログラムはまだ作っていませんから「File not found」のエラーになりますが、気にせずに次へ進みましょう。データ書き込みの確認は次章で行います。

4・5　住所録の修正

住所録データの登録が終りました。次に、データの修正プログラムを作成してみましょう。

画面レイアウト

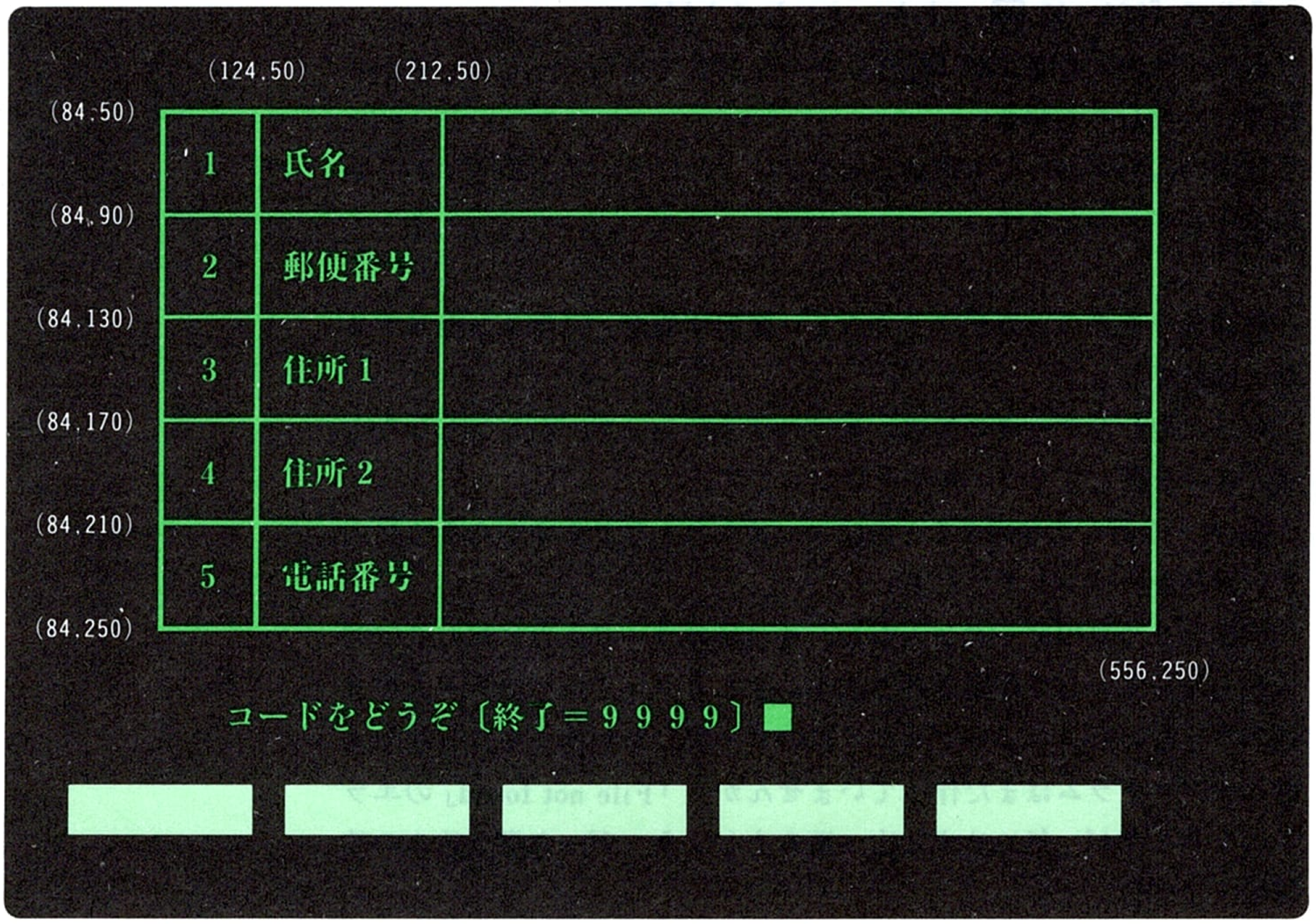
データ修正のための画面レイアウトを下に示します。

図中の座標は作表のためのグラフィック座標です。プログラミングのときの目安にしてください（グラフィック画面は640×400ドットの解像度であることに注意してください）。

図を見るとわかるように、画面下に表示されたプロンプトに従ってコード入力すると検索が行われ、もし該当コードがなければエラーメッセージを、あれば登録データを表中に出力します。

修正の場合には、修正したい項目のある枠の番号を入力し修正作業を行います。

くわしくは、あとで実際にやってみますので、そのときに覚えることにしましょう。いまは画面レイアウトの感じだけつかんでおいてください。



初期設定と前処理

この部分のプログラムを示しますので入力してください。

```
100 '  住所録プログラム
110 '  save "2:adres2.bas"
120 '  住所録メンテナンス
130 '
140 *SHOKISETTEI
150  SCREEN 4
160  WIDTH 80,20 : CONSOLE 0,20,1,0
170  DEFINT A-Z
180  DIM CODE(500),HEAD$(500)
190 '
200 *JYUNBI
210  GOSUB *SAKUHYOU    :' 作表処理
220  GOSUB *TITLE       :' タイトル表示
230  GOSUB *KOUMOKUMEI  :' 項目名表示
240  GOSUB *KENSAKUHOU  :' 検索方法の選択
250  GOSUB *MESSAGE1    :' メッセージ表示
260  GOSUB *FILE.OPEN   :' ランダムファイルのオープン
270  GOSUB *SAKUIN.OPEN :' 索引ファイルのオープン
280  GOSUB *DATA.YOMU   :' 索引を読む
290  GOSUB *GAMEN.KESU  :' 画面の一部消去
```

140〜180行が初期設定です。

200行からは前準備のためのサブルーチンへのジャンプです。1つずつサブルーチンを示していきますので入力してください。

作表サブルーチン

このルーチンは次のとおりです。

```
410 '
420 *SAKUHYOU
430  LINE (84,50)-(556,250),,B
440  LINE (84, 90)-(556, 90)
450  LINE (84,130)-(556,130)
460  LINE (84,170)-(556,170)
470  LINE (84,210)-(556,210)
480  LINE (84,250)-(556,250)
490  LINE (124,50)-(124,250)
500  LINE (212,50)-(212,250)
```

LINE文を使って前ページに示したような表を作ります。N88日本語BASICモードでは作表のためのグラフィックキャラクタが使えませんので、LINE文に頼らざるをえません。

430行のLINE文のみ「**Bオプション**」の指定がありますので、2つ

の点を結ぶ直線を対角線とする長方形を作っています。その他はすべて直線を引くだけです。

座標については224ページの図を参考にしてください。

タイトル・項目名の表示

次のプログラムを入力してください。

```
510 '
520 *TITLE
530  LOCATE 11,1 : PRINT "【住所録メンテナンス】"
540  RETURN
550 '
560 *KOUMOKUMEI
570  LOCATE 12,3 : PRINT "1"
580  LOCATE 12,5 : PRINT "2"
590  LOCATE 12,7 : PRINT "3"
600  LOCATE 12,9 : PRINT "4"
610  LOCATE 12,11: PRINT "5"
620  LOCATE 17,3 : PRINT "氏名"
630  LOCATE 17,5 : PRINT "郵便番号"
640  LOCATE 17,7 : PRINT "住所1"
650  LOCATE 17,9 : PRINT "住所2"
660  LOCATE 17,11: PRINT "電話番号"
670  RETURN
```

このルーチンは別に問題ないですね。

検索方法の選択ルーチン

次は、修正データ検索のためのキーデータをコードにするか姓の先頭1字にするかの選択ルーチンです。

```
910 '
920 *KENSAKUHOU
930  LOCATE 15,14
940  PRINT "検索方法を1か2で選んでください。"
950  LOCATE 17,16 : PRINT "1・・コード検索"
960  LOCATE 17,17 : PRINT "2・・頭文字での検索"
970  I$=INKEY$ : IF I$="" THEN GOTO 970
980  IF I$="1" THEN KOUDO.KENSAKU=-1 : GOTO 1010
990  IF I$="2" THEN KOUDO.KENSAKU=0  : GOTO 1010
1000  BEEP : BEEP : GOTO 970
1010  GOSUB *GAMEN.KESU
1020  RETURN
```

画面下部に２つのメニューが表示され、１か２の番号で入力します。１番なら**コード検索**、２番なら**頭文字での検索**となります。

980行、990行はコード検索かそうでないかをチェックするための**フラグセット**です。

画面の一部消去ルーチン

検索方法の選択が終了すると、いま表示された検索方法選択のためのメッセージを消去します。これは1010行で行っています。

この部分のサブルーチンは次のとおりです。

```
1030 ′
1040 *GAMEN.KESU
1050  FOR I=14 TO 18
1060    LOCATE 0,I : PRINT SPC(79)
1070  NEXT I
1080  RETURN
```

ここで、ついでですから、修正データ入力行の消去ルーチンもキーインしてしまいましょう。

次をキーインしてください。

```
1090 ′
1100 *NYUURYOKUGYOU.KESU
1110  LOCATE 15,14 : PRINT SPC(60) :′ 入 力 行 の 消 去
1120  RETURN
```

メッセージ出力

次に、ファイルをオープンして索引の読み出しに入りますが、この旨オペレータに知らせるためのメッセージを出力しておきます。

```
1130 ′
1140 *MESSAGE1
1150  LOCATE 15,14
1160  PRINT "デ ー タ 読 み 出 し 中 。 少 々 お 待 ち く だ さ い 。"
1170  RETURN
```

ファイルのオープン

ここでランダムファイル、シーケンシャルファイルをオープンして使える状態にしておきましょう。

```
680 '
690 *FILE.OPEN
700  OPEN "2:ｼﾞｭｳｼｮ" AS #1
710  FIELD #1,20 AS NM$,7 AS PN$,40 AS AD1$,
               40 AS AD2$,15 AS TL$
720  RETURN
730 '
740 *SAKUIN.OPEN
750  OPEN "2:ｻｸｲﾝ1" FOR INPUT AS #2   :' コード索引
760  OPEN "2:ｻｸｲﾝ2" FOR INPUT AS #3   :' 頭文字索引
770  RETURN
```

入力の際、バッファ番号を間違えないように注意してください。

索引データの読み込みルーチン

次に、すでに登録ずみのデータのコード番号および姓の先頭1文字データの読み込みです。

```
780
790 *DATA.YOMU
800  WHILE NOT EOF(2)
810    C=C+1                  :' カウンタを1進める
820    INPUT #2,CODE(C)       :' コードを1件読み出す
830  WEND
840  CLOSE #2 : C=0
850  WHILE NOT EOF(3)
860    C=C+1                  :' カウンタを1進める
870    INPUT #3,HEAD$(C)      :' 頭文字を1件読み出す
880  WEND
890  CLOSE #3 : MAX=C : C=0
900  RETURN
```

配列CODEおよびHEAD＄の中にコード番号、姓の先頭1文字を格納します。

890行のMAXは、最後に姓の頭文字用ファイルを書きなおすときのためにデータ登録数を一時的に待避させておくための変数です。つまり、これを使ってループを組み、シーケンシャルファイルを新しく書きなおします。

メインルーチン

以上で、細かいサブルーチンの入力が終りました。メインルーチンを入力してしまいましょう。メインルーチンといっても、ごく簡単なものです。

```
300 '
310 *MAIN.LOOP
320  IF KOUDO.KENSAKU THEN GOSUB *KOUDO.KENSAKU
                        ELSE GOSUB *KASHIRAMOJI.KENSAKU
```

コード検索の場合は**コード検索サブルーチン**へ、そうでない場合は**頭文字検索サブルーチン**へジャンプします。処理の流れは、それぞれのサブルーチンとメインルーチンの間をループする形を取ります。

コード検索かどうかの条件判断は、変数KOUDO.KENSAKUの内容が真か偽か（－1か0か）によっています。

では次に、コード検索および頭文字検索のサブルーチンを入力しておきましょう。プログラムを次に示します。

```
1180 '
1190 *KOUDO.KENSAKU
1200  LOCATE 15,14
1210  INPUT"コードをどうぞ[終了=9999]",CD
1220  IF CD=9999 THEN GOTO *OWARI
1230  S=SEARCH(CODE,CD)          :'コードをサーチ
1240  IF S=-1 THEN GOSUB *MITOUROKU : RETURN *MAIN.LOOP
1250  SEC=S : GOSUB *SHUUSEI : RETURN *MAIN.LOOP
1260 '
1270 *KASHIRAMOJI.KENSAKU
1280  LOCATE 15,14
1290  INPUT"頭文字をどうぞ[終了=RET]",K$
1300  IF K$="" THEN GOTO *OWARI
1310  FOR I=1 TO MAX
1320    IF HEAD$(I)=K$ THEN C=1 : SEC=I : GOSUB *SHUUSEI
1330  NEXT I
1340  IF C=0 THEN GOSUB *MITOUROKU
1350  C=0 : RETURN *MAIN.LOOP
```

コード検索の場合はコード入力の際に9999がキーインされると、また頭文字検索の場合には頭文字入力の際に ⏎ キーが押されると修正処理終了ということで終了処理へジャンプします。

登録コードがあるかどうかは**SEARCH関数**で、また該当の頭文字があるかどうかは配列HEAD$の中をFOR～NEXTループを組んで探します。いずれの場合にも、該当データがあれば修正ルーチンへ飛ばし

ます。

1320行で該当する頭文字があった場合C＝1を行っていますが、これは1340行で未登録サブルーチンへジャンプさせるかどうかをチェックするためです。

修正ルーチン

次に修正ルーチン、続いて未登録のエラーメッセージ出力ルーチンを示しますので入力してください。

```
1380  GOSUB *NYUURYOKUGYOU.KESU
1390  GET #1,SEC
1400  LOCATE 28,3 : PRINT NM$    :' 氏 名 の 表 示
1410  LOCATE 28,5 : PRINT PN$    :' 郵 便 番 号 の 表 示
1420  LOCATE 28,7 : PRINT AD1$   :' 住 所 1 の 表 示
1430  LOCATE 28,9 : PRINT AD2$   :' 住 所 2 の 表 示
1440  LOCATE 28,11: PRINT TL$    :' 電 話 番 号 の 表 示
1450 *SHUUSEI.LOOP
1460  GOSUB *NYUURYOKUGYOU.KESU
1470  FOR BUZZER=1 TO 100 : BEEP 1 : BEEP 0 : NEXT
1480  LOCATE 15,14
1490  PRINT "修 正 項 目 番 号 を ど う ぞ [ 終 了 = R E T ] ";
1500  I$=INKEY$ : IF I$="" THEN GOTO 1500
1510  IF I$=CHR$(13) THEN GOTO *DEGUCHI
1520  N=VAL(I$)
1530  IF N<1 OR N>5 THEN BEEP : GOTO 1480
1540  ON N GOTO *NAMAE,*YUUBIN,*JUUSHO1,*JUUSHO2,*DENWA
1550 *NAMAE
1560  GOSUB *NYUURYOKUGYOU.KESU
1570  LOCATE 25,14 : PRINT NM$
1580  LOCATE 15,14 : INPUT "氏 名    ",NAMAE$
1590  IF NAMAE$="" THEN BEEP : GOTO 1580
1600  LOCATE 28,3  : PRINT SPC(40)
1610  LOCATE 28,3  : PRINT NAMAE$   :' 旧データ表示
1620  HEAD$(SEC)=LEFT$(NAMAE$,2)  :' 頭 文 字 の 差 し 替 え
1630  LSET NM$=NAMAE$             :' 新 デ ー タ を セ ッ ト
1640  GOTO *SHUUSEI.LOOP          :' 修 正 ル ー プ へ 戻 る
1650 *YUUBIN
1660  GOSUB *NYUURYOKUGYOU.KESU
1670  LOCATE 25,14 : PRINT PN$
1680  LOCATE 15,14 : INPUT "郵 便 番 号  ",YUUBIN$
1690  LOCATE 28, 5 : PRINT SPC(40)
1700  LOCATE 28, 5 : PRINT YUUBIN$
1710  LSET PN$=YUUBIN$
1720  GOTO *SHUUSEI.LOOP
1730 *JUUSHO1
1740  GOSUB *NYUURYOKUGYOU.KESU
1750  LOCATE 25,14 : PRINT AD1$
```

```
1760  LOCATE 15,14 : INPUT "住所1     ",JUUSHO1$
1770  LOCATE 28, 7 : PRINT SPC(40)
1780  LOCATE 28, 7 : PRINT JUUSHO1$
1790  LSET AD1$=JUUSHO1$
1800  GOTO *SHUUSEI.LOOP
1810 *JUUSHO2
1820  GOSUB *NYUURYOKUGYOU.KESU
1830  LOCATE 25,14 : PRINT AD2$
1840  LOCATE 15,14 : INPUT "住所2     ",JUUSHO2$
1850  LOCATE 28, 9 : PRINT SPC(40)
1860  LOCATE 28, 9 : PRINT JUUSHO2$
1870  LSET AD2$=JUUSHO2$
1880  GOTO *SHUUSEI.LOOP
1890 *DENWA
1900  GOSUB *NYUURYOKUGYOU.KESU
1910  LOCATE 25,14 : PRINT TL$
1920  LOCATE 15,14 : INPUT "電話番号  ",DENWA$
1930  LOCATE 28,11 : PRINT SPC(40)
1940  LOCATE 28,11 : PRINT DENWA$
1950  LSET TL$=DENWA$
1960  GOTO *SHUUSEI.LOOP
1970 *DEGUCHI
1980  GOSUB *NYUURYOKUGYOU.KESU
1990  PUT #1,SEC         :' 新データをファイルに書く
2000  FOR J=1 TO 5
2010    LOCATE 28,2*J+1 : PRINT SPC(40)
2020  NEXT J
2030  RETURN
```

```
2040 '
2050 *MITOUROKU
2060  BEEP : BEEP
2070  GOSUB *NYUURYOKUGYOU.KESU
2080  LOCATE 15,14
2090  PRINT "登録してありません!!"
2100  FOR TIMER=1 TO 1000 : NEXT TIMER
2110  GOSUB *NYUURYOKUGYOU.KESU
2120  RETURN
```

修正ルーチンの原理は、修正すべき登録データをランダムファイルより取り出し、それぞれの項目を画面に表示されている表の中に出力し、修正項目番号をキーインすることによって行います。

修正項目番号の入力促進に対して ⏎ キーが押されると、現在画面に表示されているレコードに対しての修正作業は終りということで修正ループの出口へジャンプし、ランダムファイルに新データを書き込んで再びメインループへ飛びます。

プログラムの流れは複雑ではありませんから、追っていただけば理

解できると思います。

処理終了ルーチン

最後に、終りの処理を入力しましょう。変更のあったシーケンシャルファイルを新しく書きなおすのです。

```
330 '
340 *OWARI
350  OPEN "2:ｻｸｲﾝ2" FOR OUTPUT AS #2
360  FOR I=1 TO MAX
370    WRITE #2,HEAD$(I)
380  NEXT I
390  CLOSE
400  RUN "2:adres4.bas"   :' メニューへ戻る
```

ファイルを閉じたらメニュープログラムを実行します。

次ページ以降に、「住所録メンテナンスプログラム」の総リストをかかげておきますので、正常に動作しなかった場合、プリンタにリストを打ち出しバグ探しにお使いください。

```
100 ' 住所録プログラム
110 ' save "2:adres2.bas"
120 ' 住所録メンテナンス
130 '
140 *SHOKISETTEI
150  SCREEN 4
160  WIDTH 80,20 : CONSOLE 0,20,1,0
170  DEFINT A-Z
180  DIM CODE(500),HEAD$(500)
190 '
200 *JYUNBI
210  GOSUB *SAKUHYOU     :' 作表処理
220  GOSUB *TITLE        :' タイトル表示
230  GOSUB *KOUMOKUMEI   :' 項目名表示
240  GOSUB *KENSAKUHOU   :' 検索方法の選択
250  GOSUB *MESSAGE1     :' メッセージ表示
260  GOSUB *FILE.OPEN    :' ランダムファイルのオープン
270  GOSUB *SAKUIN.OPEN  :' 索引ファイルのオープン
280  GOSUB *DATA.YOMU    :' 索引を読む
290  GOSUB *GAMEN.KESU   :' 画面の一部消去
300 '
310 *MAIN.LOOP
320  IF KOUDO.KENSAKU THEN GOSUB *KOUDO.KENSAKU
                      ELSE GOSUB *KASHIRAMOJI.KENSAKU
330 '
340 *OWARI
350  OPEN "2:ｻｸｲﾝ2" FOR OUTPUT AS #2
360  FOR I=1 TO MAX
370    WRITE #2,HEAD$(I)
380  NEXT I
390  CLOSE
400  RUN "2:adres4.bas"       :' メニューへ戻る
410 '
420 *SAKUHYOU
430  LINE (84,50)-(556,250),,B
440  LINE (84, 90)-(556, 90)
450  LINE (84,130)-(556,130)
460  LINE (84,170)-(556,170)
470  LINE (84,210)-(556,210)
480  LINE (84,250)-(556,250)
490  LINE (124,50)-(124,250)
500  LINE (212,50)-(212,250)
510 '
520 *TITLE
530  LOCATE 11,1 : PRINT "【住所録メンテナンス】"
540  RETURN
550 '
560 *KOUMOKUMEI
570  LOCATE 12,3 : PRINT "1"
580  LOCATE 12,5 : PRINT "2"
590  LOCATE 12,7 : PRINT "3"
600  LOCATE 12,9 : PRINT "4"
610  LOCATE 12,11: PRINT "5"
620  LOCATE 17,3 : PRINT "氏名"
```

```
630  LOCATE 17,5 : PRINT "郵便番号"
640  LOCATE 17,7 : PRINT "住所1"
650  LOCATE 17,9 : PRINT "住所2"
660  LOCATE 17,11: PRINT "電話番号"
670  RETURN
680 '
690 *FILE.OPEN
700  OPEN "2:ｼﾞｭｳｼｮ" AS #1
710  FIELD #1,20 AS NM$,7 AS PN$,40 AS AD1$,
              40 AS AD2$,15 AS TL$
720  RETURN
730 '
740 *SAKUIN.OPEN
750  OPEN "2:ｻｸｲﾝ1" FOR INPUT AS #2  :' コード索引
760  OPEN "2:ｻｸｲﾝ2" FOR INPUT AS #3  :' 頭文字索引
770  RETURN
780 '
790 *DATA.YOMU
800  WHILE NOT EOF(2)
810    C=C+1                :' カウンタを1進める
820    INPUT #2,CODE(C)     :' コードを1件読み出す
830  WEND
840  CLOSE #2 : C=0
850  WHILE NOT EOF(3)
860    C=C+1                :' カウンタを1進める
870    INPUT #3,HEAD$(C)    :' 頭文字を1件読み出す
880  WEND
890  CLOSE #3 : MAX=C : C=0
900  RETURN
910 '
920 *KENSAKUHOU
930  LOCATE 15,14
940  PRINT "検索方法を1か2で選んでください。"
950  LOCATE 17,16 : PRINT "1・・コード検索"
960  LOCATE 17,17 : PRINT "2・・頭文字での検索"
970  I$=INKEY$ : IF I$="" THEN GOTO 970
980  IF I$="1" THEN KOUDO.KENSAKU=-1 : GOTO 1010
990  IF I$="2" THEN KOUDO.KENSAKU=0  : GOTO 1010
1000  BEEP : BEEP : GOTO 970
1010  GOSUB *GAMEN.KESU
1020  RETURN
1030 '
1040 *GAMEN.KESU
1050  FOR I=14 TO 18
1060    LOCATE 0,I : PRINT SPC(79)
1070  NEXT I
1080  RETURN
1090 '
1100 *NYUURYOKUGYOU.KESU
1110  LOCATE 15,14 : PRINT SPC(60) :' 入力行の消去
1120  RETURN
1130 '
1140 *MESSAGE1
1150  LOCATE 15,14
```

```
1160  PRINT "データ読み出し中。少々お待ちください。"
1170  RETURN
1180 '
1190 *KOUDO.KENSAKU
1200  LOCATE 15,14
1210  INPUT "コードをどうぞ[終了=9999] ",CD
1220  IF CD=9999 THEN GOTO *OWARI
1230  S=SEARCH(CODE,CD)          :' コードをサーチ
1240  IF S=-1 THEN GOSUB *MITOUROKU : RETURN *MAIN.LOOP
1250  SEC=S : GOSUB *SHUUSEI : RETURN *MAIN.LOOP
1260 '
1270 *KASHIRAMOJI.KENSAKU
1280  LOCATE 15,14
1290  INPUT "頭文字をどうぞ[終了=RET] ",K$
1300  IF K$="" THEN GOTO *OWARI
1310  FOR I=1 TO MAX
1320    IF HEAD$(I)=K$ THEN C=1 : SEC=I : GOSUB *SHUUSEI
1330  NEXT I
1340  IF C=0 THEN GOSUB *MITOUROKU
1350  C=0 : RETURN *MAIN.LOOP
1360 '
1370 *SHUUSEI
1380  GOSUB *NYUURYOKUGYOU.KESU
1390  GET #1,SEC
1400  LOCATE 28,3 : PRINT NM$    :' 氏名の表示
1410  LOCATE 28,5 : PRINT PN$    :' 郵便番号の表示
1420  LOCATE 28,7 : PRINT AD1$   :' 住所1の表示
1430  LOCATE 28,9 : PRINT AD2$   :' 住所2の表示
1440  LOCATE 28,11: PRINT TL$    :' 電話番号の表示
1450 *SHUUSEI.LOOP
1460  GOSUB *NYUURYOKUGYOU.KESU
1470  FOR BUZZER=1 TO 100 : BEEP 1 : BEEP 0 : NEXT
1480  LOCATE 15,14
1490  PRINT "修正項目番号をどうぞ[終了=RET]";
1500  I$=INKEY$ : IF I$="" THEN GOTO 1500
1510  IF I$=CHR$(13) THEN GOTO *DEGUCHI
1520  N=VAL(I$)
1530  IF N<1 OR N>5 THEN BEEP : GOTO 1480
1540  ON N GOTO *NAMAE,*YUUBIN,*JUUSHO1,*JUUSHO2,*DENWA
1550 *NAMAE
1560  GOSUB *NYUURYOKUGYOU.KESU
1570  LOCATE 25,14 : PRINT NM$
1580  LOCATE 15,14 : INPUT "氏名       ",NAMAE$
1590  IF NAMAE$="" THEN BEEP : GOTO 1580
1600  LOCATE 28,3  : PRINT SPC(40)
1610  LOCATE 28,3  : PRINT NAMAE$   :' 旧データ表示
1620  HEAD$(SEC)=LEFT$(NAMAE$,2)   :' 頭文字の差し替え
1630  LSET NM$=NAMAE$              :' 新データをセット
1640  GOTO *SHUUSEI.LOOP           :' 修正ループへ戻る
1650 *YUUBIN
1660  GOSUB *NYUURYOKUGYOU.KESU
1670  LOCATE 25,14 : PRINT PN$
1680  LOCATE 15,14 : INPUT "郵便番号  ",YUUBIN$
1690  LOCATE 28, 5 : PRINT SPC(40)
```

```
1700  LOCATE 28, 5 : PRINT YUUBIN$
1710  LSET PN$=YUUBIN$
1720  GOTO *SHUUSEI.LOOP
1730 *JUUSHO1
1740  GOSUB *NYUURYOKUGYOU.KESU
1750  LOCATE 25,14 : PRINT AD1$
1760  LOCATE 15,14 : INPUT "住所 1      ",JUUSHO1$
1770  LOCATE 28, 7 : PRINT SPC(40)
1780  LOCATE 28, 7 : PRINT JUUSHO1$
1790  LSET AD1$=JUUSHO1$
1800  GOTO *SHUUSEI.LOOP
1810 *JUUSHO2
1820  GOSUB *NYUURYOKUGYOU.KESU
1830  LOCATE 25,14 : PRINT AD2$
1840  LOCATE 15,14 : INPUT "住所 2      ",JUUSHO2$
1850  LOCATE 28, 9 : PRINT SPC(40)
1860  LOCATE 28, 9 : PRINT JUUSHO2$
1870  LSET AD2$=JUUSHO2$
1880  GOTO *SHUUSEI.LOOP
1890 *DENWA
1900  GOSUB *NYUURYOKUGYOU.KESU
1910  LOCATE 25,14 : PRINT TL$
1920  LOCATE 15,14 : INPUT "電話番号    ",DENWA$
1930  LOCATE 28,11 : PRINT SPC(40)
1940  LOCATE 28,11 : PRINT DENWA$
1950  LSET TL$=DENWA$
1960  GOTO *SHUUSEI.LOOP
1970 *DEGUCHI
1980  GOSUB *NYUURYOKUGYOU.KESU
1990  PUT #1,SEC           : 新データをファイルに書く
2000  FOR J=1 TO 5
2010    LOCATE 28,2*J+1 : PRINT SPC(40)
2020  NEXT J
2030  RETURN
2040 '
2050 *MITOUROKU
2060  BEEP : BEEP
2070  GOSUB *NYUURYOKUGYOU.KESU
2080  LOCATE 15,14
2090  PRINT "登録してありません！！"
2100  FOR TIMER=1 TO 1000 : NEXT TIMER
2110  GOSUB *NYUURYOKUGYOU.KESU
2120  RETURN
```

4・6　実際に修正してみよう

このプログラムを実行すると次ページの上の図のように項目名の入った表が現れ、画面下部に、コードによる検索か氏名の頭文字による検索かを選択するようメッセージが表示されますから、どちらかを選んでください。

ここでは、たとえば1を選択してみましょう。

すると、画面に

データ読み出し中です。少々お待ちください。

のメッセージが表示され、しばらくして、

コードをどうぞ［終了＝9999］

というプロンプト文が表示されます。

たとえばコード番号100番の福島実さんの修正をすることにして

100 ⏎

とすると、画面は次ページの下の図のようになり修正項目番号をたずねてきます。

各自適当な番号を入力して修正を行ってみてください。

たとえば「住所2」を変更したいとしますと、項目番号は4ですから「4 ⏎」とキーインします。

すると、画面に旧データが表示され、その先頭でカーソルがブリンクしています。f・1 キーを押して日本語入力モードにし、「1－2－3 ⏎」などと入力してみてください。これで新しい住所2のデータが決定されますから、もう一度 ⏎ キーを押します。

今度はブザーが鳴り、表中の古いデータが新しいデータと入れ換わります。画面下には再び項目番号入力のプロンプトが表示されますから、もっと修正したければ同様に行ってみてください。

修正を終りたければ ⏎ キーを押します。するとコード番号の入力になりますから、修正処理を完全に終了したければ「9999」をキーインしてください。

メニュープログラムが入力してないのでエラーメッセージが表示されますが、気にせずに、もう一度プログラムを実行し、いま修正したデータを呼び出して正しく変更されたかどうか確認してみてください。

【住所録メンテナンス】

1	氏名	
2	郵便番号	
3	住所 1	
4	住所 2	
5	電話番号	

検索方法を 1 か 2 で選んでください。
　1・・コード検索
　2・・頭文字での検索

1	氏名	福島　実
2	郵便番号	962－03
3	住所 1	福島県岩瀬郡岩瀬村
4	住所 2	100番地
5	電話番号	111－111－1111

修正項目番号をどうぞ［終了＝RET］

4·7 住所録データの印刷

私たちは第１章で宛て名印刷を体験しましたが、このときはカタカナ（１バイト系文字列）の出力でした。本節では漢字**データを宛て名印刷形式で出力**してみます。

また第１章では、１名ずつ縦方向に打ち出しただけでしたので、プリンタ用紙の右側部分が少しムダになってしまいました。そこで本節では横方向に２名ずつ印刷し、少しばかり節約をしようと思います。

まず、ファイルオープンまでのプログラムを次に示します。

```
100 '  住所録プログラム
110 '  save "2:adres3.bas"
120 '  住所録の印刷
130 '
140 *SHOKISETTEI
150  SCREEN 4
160  WIDTH 80,25 : CONSOLE 0,25,1,0
170  DEFINT A-Z
180 '
190 *MESSAGE.HYOUJI
200  LOCATE 30,5
210  PRINT "印刷中"
220  FOR TIMER=1 TO 1000 : NEXT TIMER
230 '
240 *FILE.OPEN
250  OPEN "2:ｼﾞｭｳｼｮ" AS #1
260  FIELD #1,20 AS NM$,7 AS PN$,40 AS AD1$,
           40 AS AD2$,15 AS TL$
270  MAX=LOF(1)
```

220行のタイマーは、印刷件数が少ないとアッという間に終ってしまいますので、処理過程のバランスをとる意味で入れておきました。

印刷件数は270行でやっているように**LOF関数**によって与えられます。

次に印刷部分のプログラムを次ページに示します。

横方向に２名分印刷しますので、**変数Ａ１＄〜Ａ４＄**および**変数Ｂ１＄〜Ｂ４＄**を用意しています。そして、バッファ上に取り出されたデータをこれらの変数に代入するごとに**変数MAX**の内容を１ずつ減らしています。これはどういうことかといいますと、データ終りのチェックをするためです。つまり、１ずつ減らしていってMAXの内容が０になったとき、**変数OWARI**に**－１（真）**を代入し、終りのフラグを立てておきます。

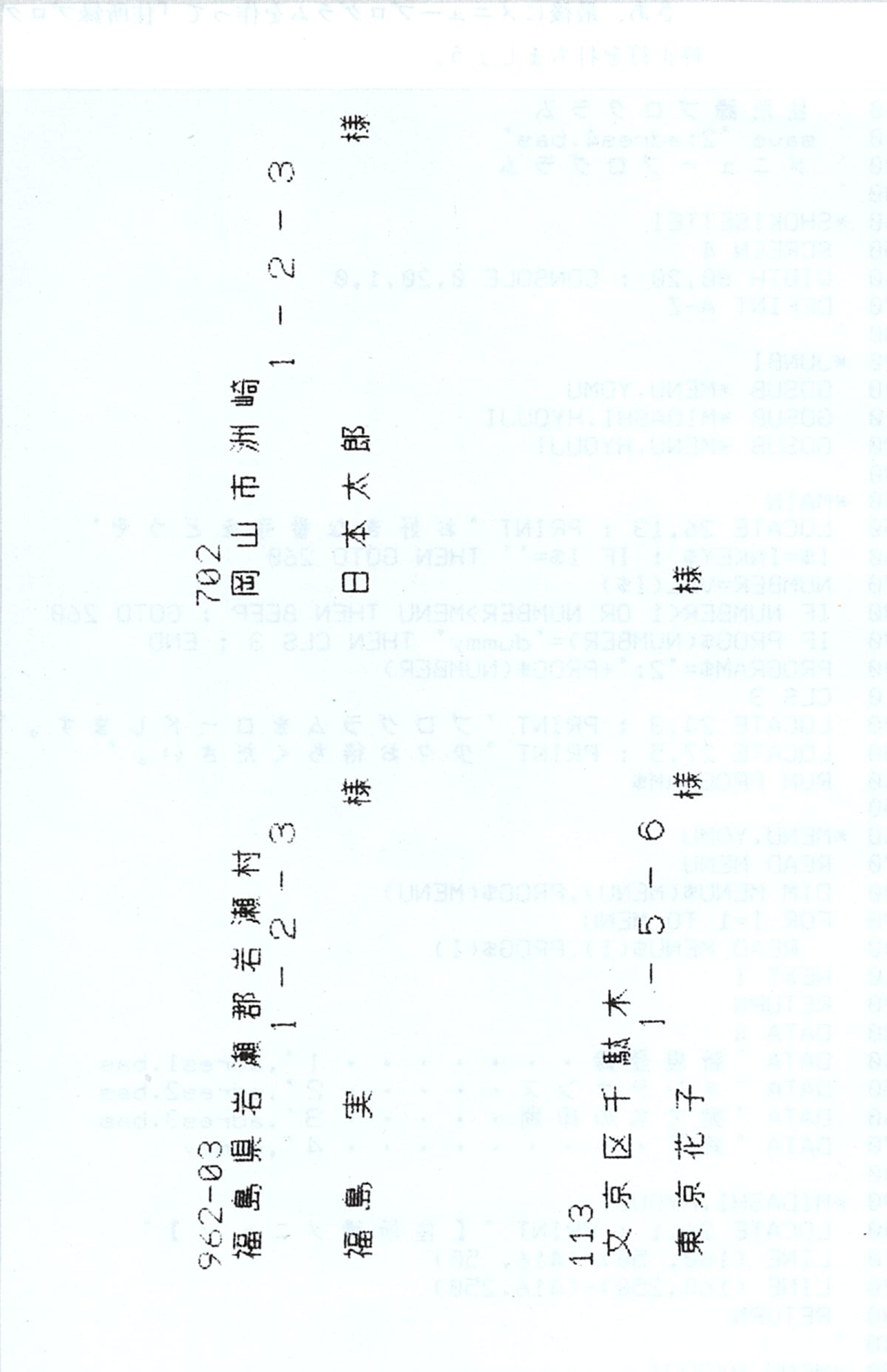
```
280 '
290 *LOOP
300  MAX=MAX-1
310  SEC=SEC+1
320  GET #1,SEC
330  A1$=NM$  : A2$=PN$
340  A3$=AD1$ : A4$=AD2$
350  IF MAX=0 THEN OWARI=-1 : GOTO *INSATSU
360  MAX=MAX-1
370  IF MAX=0 THEN OWARI=-1
380  SEC=SEC+1
390  GET #1,SEC
400  B1$=NM$  : B2$=PN$
410  B3$=AD1$ : B4$=AD2$
420 *INSATSU
430  LPRINT A2$;CHR$(13);TAB(30);B2$
440  LPRINT A3$;CHR$(13);TAB(30);B3$
450  LPRINT TAB(10);A4$;CHR$(13);TAB(40);B4$
460  LPRINT A1$;"様";CHR$(13);TAB(30);B1$;"様"
470  IF LC=5 THEN LC=0 : LPRINT CHR$(12);
480  FOR I=1 TO 5 : LPRINT : NEXT I
490  IF OWARI THEN CLOSE : RUN "2:adres4.bas"
500  B1$="" : B2$="" : B3$="" : B4$=""
510  GOTO *LOOP
```

430行の**ＬＣ**は**カウンタ**です。縦方向に５件、ということは横方向に２件印刷しますから計10件を印刷するごとに改ページ処理を行います。改ページは480行で行っています。**CHR＄（12）**を出力すればよいですね。

440行の「**CHR$(13);**」に注意してください。これは**キャリッジリターン**のキャラクタコード「**13**」を出力するものですが、うしろにセミコロン「；」をつけることにより、印字ヘッドが改行されずに、その行の先頭に合わせられます。

具体的にいいますと、440行ではまず最初の人の郵便番号が出力され、次に印字ヘッドがもう一度、最初の人の郵便番号の先頭に合わされ、次に30字分（１バイト系）のタブが出力され、そのうしろに２番目の人の郵便番号が出力されます。

プログラムが入力できたらセーブしてから実行してみてください。次ページのようにプリンタに印字されます。


702
岡山市洲崎 1-2-3
日本太郎 様
962-03
福島県岩瀬郡若瀬村 1-2-3
福島実 様
113
文京区千駄木 1-5-6
東京花子 様

4・8 住所録メニュープログラム

さあ、最後にメニュープログラムを作って「住所録プログラム」に終止符を打ちましょう。

```
100 ' 住所録プログラム
110 ' save "2:adres4.bas"
120 ' メニュープログラム
130 '
140 *SHOKISETTEI
150  SCREEN 4
160  WIDTH 80,20 : CONSOLE 0,20,1,0
170  DEFINT A-Z
180 '
190 *JUNBI
200  GOSUB *MENU.YOMU
210  GOSUB *MIDASHI.HYOUJI
220  GOSUB *MENU.HYOUJI
230 '
240 *MAIN
250  LOCATE 26,13 : PRINT "お好きな番号をどうぞ"
260  I$=INKEY$ : IF I$="" THEN GOTO 260
270  NUMBER=VAL(I$)
280  IF NUMBER<1 OR NUMBER>MENU THEN BEEP : GOTO 260
290  IF PROG$(NUMBER)="dummy" THEN CLS 3 : END
300  PROGRAM$="2:"+PROG$(NUMBER)
310  CLS 3
320  LOCATE 24,3 : PRINT "プログラムをロードします。"
330  LOCATE 27,5 : PRINT "少々お待ちください。"
340  RUN PROGRAM$
350 '
360 *MENU.YOMU
370  READ MENU
380  DIM MENU$(MENU),PROG$(MENU)
390  FOR I=1 TO MENU
400    READ MENU$(I),PROG$(I)
410  NEXT I
420  RETURN
430  DATA 4
440  DATA "新規登録・・・・・・・・1",adres1.bas
450  DATA "メンテナンス・・・・・2",adres2.bas
460  DATA "宛て名の印刷・・・・・3",adres3.bas
470  DATA "終了・・・・・・・・・・4",dummy
480 '
490 *MIDASHI.HYOUJI
500  LOCATE 26,1 : PRINT "【住所録メニュー】"
510  LINE (160, 50)-(416, 50)
520  LINE (160,250)-(416,250)
530  RETURN
540 '
550 *MENU.HYOUJI
560  FOR I=1 TO MENU
570    LOCATE 24,I+2 : PRINT MENU$(I)
580  NEXT I
590  RETURN
```

PC-8801MKⅡFR／MR
データファイル入門2

編　者　ナツメ出版企画
発行者　田村正隆
発行所　株式会社　ナツメ社
東京都千代田区神田神保町1－52(〒101)
電話　03(291)1257(代表)
振替　東京3－58661

印　刷　ラン印刷社
製　本　三修紙工

ISBN4-8163-0567-X

PC-8801MKⅡFR/MR

プログラミング＆オペレーション入門

ナツメ出版企画編―――1700円

◉PC-8801MKⅡFR/MRは、従来のN88-BASIC V1はもとより、PC-8801MKⅡSRの持つアナログパレット機能、FM音源によるサウンド機能に加えて、さらに手軽に漢字が扱えるN88日本語BASICをも追加した多機能型パソコンで、ファミリーユースとしてもビジネスユースとしても、とにかく幅広く多方面に活用できます。本書では、さまざまな分野に題材を求め、プログラミングの基本から応用までをやさしく解説しました。この本を学べばプログラミングに相当自信がつくはずです。

【内容】◎電源を入れてみよう◎キー操作を覚えよう◎関数計算もできます◎画面をカラーに◎プログラムのセーブとロード◎表に数値を書き込む◎配列を使おう◎プログラムの完成◎ブロックくずし◎ボールが何に当たった？◎ボール表示とブロック消去◎算数教室◎プログラムのあらすじ◎たし算のサブルーチン◎もっとチャレンジしたければ◎DMプログラム◎データサーチ◎宛て名の印刷◎拡張命令について◎サウンド機能をいかそう◎タートルグラフィックの使い方◎グラフィック画面のモード◎円を描いて中を塗りつぶす◎ワールド座標に円を描く◎絵を描いてみよう◎ドレミファソラシド◎ボリュームとテンポ◎「禁じられた遊び」のテーマ音楽◎走れSL！◎プリンタの使い方◎日本語を入力してみよう◎プログラムの中で漢字をつかおう

PC-8801MK II FR/MR

データファイル入門1

ナツメ出版企画編———1600円

◉パソコンをビジネスその他、実務面で役立てようと思うと、どうしてもディスク操作のマスターをせまられます。しかしディスクでのデータの扱いは、ベーシックの場合と多少考え方を異にするためすぐになじめない面があります。本書ではベーシックとディスクベーシックとの連絡をつけ、できるだけ抵抗なくディスク操作が体得できるように解説しました。

【内容】◎システムディスクのコピーを作る◎データディスクを作る◎ファイルの転送◎ファイル名の変更と削除◎ディスクの構造◎ディスクにデータを書き込む◎ディスクのデータを読む◎電話帳ファイルの作成◎データの検索◎データの修正◎データを印刷する◎データの並べかえ◎ランダムファイルの書き込み◎ランダムファイルのデータを読む◎宛て名印刷プログラム◎マスターメニュープログラム

PC-8801MKⅡFR/MR
データファイル入門2

ナツメ出版企画編———1600円

◉本書は「PC-8801MKⅡFR/MR データファイル入門1」の続編です。前編で扱ったテーマに関連性をもたせ、みなさんが知らず知らずのうちに、より高いレベルへ進めるように工夫してあります。日本語による住所録プログラムにもチャレンジしてみます。

【内容】◎ファイル内でのデータ設計◎データの書き込み◎書き込みデータの確認◎データの修正◎宛て名の印刷◎通信文を印刷する◎ファイルに、より資料性を◎データ書き込みのプログラム◎果たしてうまく書き込めたか◎プログラムの実行と確認◎データ更新のプログラム◎さらに汎用性のある宛て名印刷◎漢字住所録プログラム◎新規登録プログラム◎住所録の修正◎住所録データの印刷

PC-8801MKII＋SR＋TR

実用版・ディスク実践プログラム集

安立隆一著―――1600円

◉本書には、キーインするだけで即使える実用的なプログラム7本のリストとその使い方が掲載されています。扱ったプログラムは、「512色中36色イラスト作画プログラム」「成績管理プログラム」「成績順ソートプログラム」「スクリーン・ロードセーブ・プログラム」「顧客管理プログラム」「名前ソートプログラム」「スケジュール管理プログラム」です。

◉本書に掲載したプログラムをすべて収録したディスケットサービスも行っています。媒体は5インチ両面倍密度ディスケット（5インチ2D）です。定価は8000円、送料350円です。ご注文の折は、「PC-8801MKII＋SR＋TR 実用版・ディスク実践プログラム集専用ディスケット」と明記してください。

【内容】◎512色中36色イラスト作画プログラム◎成績管理プログラム◎使用システムと立ち上げ◎科目名の登録◎人名リストの出力◎得点リストの出力◎得点順位◎平均・最高・最低得点◎成績順に並べかえ◎グラフィック画面の保存◎顧客管理プログラム◎顧客名の登録◎売り上げ金額の入力◎入金の登録◎売り上げ金額の合計◎アイウエオ順に並べかえ◎スケジュール管理プログラム◎予定の入力◎予定の自動作成◎時刻早送り◎割り込みプログラム

PC-8801/mkII/SR/TR/FR
漢字BASIC入門

戸内順一著――――――――――1800円

◉従来PC-8801／mkII／SR／TRでは、実務に耐えられるような日本語の漢字処理が不可能でした。本書は［8801漢字BASIC（システムソフト発売）や［N88-日本語BASIC］（日本電気製）を使い、実務的なアプリケーション・プログラムづくりに漢字処理を活かせるよう解説しました。第3部では、ミニワープロの作り方、使い方についても解説し、漢字を使ったプログラム・テクニックが自然に身につくようにしました。

【内容】◎N88-日本語BASIC◎N88-日本語BASICの起動と準備◎日本語入力◎BASICと日本語◎N88-日本語BASICの拡張利用法◎8801漢字BASIC◎8801漢字BASICの起動と準備◎日本語入力◎ディスクファイルと日本語◎シーケンシャルファイルと日本語◎ランダムファイルと日本語◎漢字BASICの拡張利用法◎ミニワープロ◎ミニワープロを考えよう◎ミニワープロを作ろう◎ミニ日本語ワープロを使おう◎付録

PC-8801mkⅡFR | MR
データファイル入門[2]

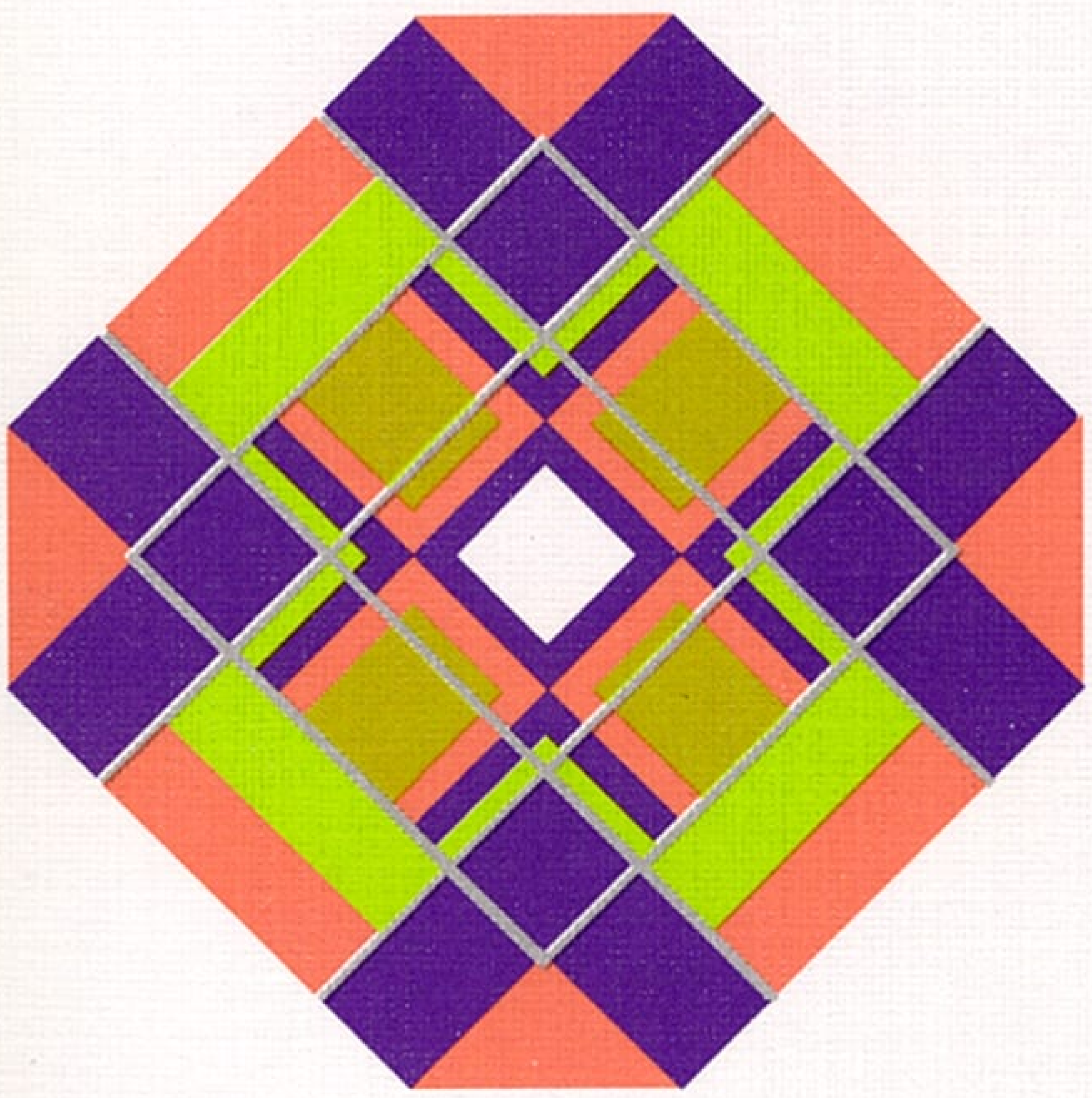
発行――――1986年3月15日
編者――――ナツメ出版企画
発行者―――田村正隆
発行所―――株式会社ナツメ社
郵便番号101
東京都千代田区神田神保町1-52
電話〈03〉291-1257
振替　東京3-58661
〈落丁・乱丁本はお取り替えします〉

定価――――1600円

PC-8801mkⅡFR|MR

データファイル入門[2]

[目次より]



ナツメ社
定価＝1600円

ISBN4-8163-0567-X C2054 ¥1600E